# Title

stata.com

**rotatemat —** Orthogonal and oblique rotations of a Stata matrix



## Description

`rotatemat` applies a linear transformation to the specified matrix so that the result minimizes a criterion function over all matrices in a class of feasible transformations. Two classes are supported: orthogonal (orthonormal) and oblique. A wide variety of criterion functions are available, representing different ways to measure the "simplicity" of a matrix. Most of these criteria can be applied with both orthogonal and oblique rotations.

This entry describes the computation engine for orthogonal and oblique transformations of Stata matrices. This command may be used directly on any Stata matrix.

If you are interested in rotation after `factor`, `factormat`, `pca`, or `pcamat`, see [MV] **factor postestimation**, [MV] **pca postestimation**, and the general description of `rotate` as a postestimation facility in [MV] **rotate**.

## Menu

Statistics > Multivariate analysis > Orthogonal and oblique rotations of a matrix

# Syntax

```
rotatemat matrix_L [ , options ]
```

| *options* | Description |
|---|---|
| **Main** | |
| orthogonal | restrict to orthogonal rotations; the default, except with promax() |
| oblique | allow oblique rotations |
| *rotation_methods* | rotation criterion |
| normalize | rotate Kaiser normalized matrix |
| **Reporting** | |
| format(%*fmt*) | display format for matrices; default is format(%9.5f) |
| blanks(#) | display loadings as blanks when \|loading\| < #; default is blanks(0) |
| nodisplay | suppress all output except log and trace |
| noloading | suppress display of rotated loadings |
| norotation | suppress display of rotation matrix |
| matname(*string*) | descriptive label of the matrix to be rotated |
| colnames(*string*) | descriptive name for columns of the matrix to be rotated |
| **Optimization** | |
| *optimize_options* | control the optimization process; seldom used |

| *rotation_methods* | Description |
|---|---|
| * varimax | varimax (orthogonal only); the default |
| vgpf | varimax via the GPF algorithm (orthogonal only) |
| quartimax | quartimax (orthogonal only) |
| equamax | equamax (orthogonal only) |
| parsimax | parsimax (orthogonal only) |
| entropy | minimum entropy (orthogonal only) |
| tandem1 | Comrey's tandem 1 principle (orthogonal only) |
| tandem2 | Comrey's tandem 2 principle (orthogonal only) |
| * promax[(#)] | promax power # (implies oblique); default is promax(3) |
| oblimin[(#)] | oblimin with $\gamma = \#$; default is oblimin(0) |
| cf(#) | Crawford–Ferguson family with $\kappa = \#$, $0 \leq \# \leq 1$ |
| bentler | Bentler's invariant pattern simplicity |
| oblimax | oblimax |
| quartimin | quartimin |
| target(*Tg*) | rotate toward matrix *Tg* |
| partial(*Tg W*) | rotate toward matrix *Tg*, weighted by matrix *W* |

* varimax and promax ignore all *optimize_options*.

# Options

## Main

`orthogonal` specifies that an orthogonal rotation be applied. This is the default.

See *Rotation criteria* below for details on the *rotation_methods* available with `orthogonal`.

`oblique` specifies that an oblique rotation be applied. This often yields more interpretable factors with a simpler structure than that obtained with an orthogonal rotation. In many applications (for example, after `factor` and `pca`), the factors before rotation are orthogonal (uncorrelated), whereas the oblique rotated factors are correlated.

See *Rotation criteria* below for details on the *rotation_methods* available with `oblique`.

`normalize` requests that the rotation be applied to the Kaiser normalization (Horst 1965) of the matrix $\mathbf{A}$ so that the rowwise sums of squares equal 1.

## Reporting

`format(%`*fmt*`)` specifies the display format for matrices. The default is `format(%9.5f)`.

`blanks(#)` specifies that small values of the rotated matrix—that is, those elements of $\mathbf{A}(\mathbf{T}')^{-1}$ that are less than # in absolute value—are displayed as spaces.

`nodisplay` suppresses all output except the log and trace.

`noloading` suppresses the display of the rotated loadings.

`norotation` suppresses the display of the optimal rotation matrix.

`matname(`*string*`)` is a rarely used output option; it specifies a descriptive label of the matrix to be rotated.

`colnames(`*string*`)` is a rarely used output option; it specifies a descriptive name to refer to the columns of the matrix to be rotated. For instance, `colnames(components)` specifies that the output label the columns as "components". The default is "factors".

## Optimization

*optimize_options* control the iterative optimization process. These options are seldom used.

`iterate(#)` is a rarely used option; it specifies the maximum number of iterations. The default is `iterate(1000)`.

`log` specifies that an iteration log be displayed.

`trace` is a rarely used option; it specifies that the rotation be displayed at each iteration.

`tolerance(#)` is one of three criteria for declaring convergence and is rarely used. The `tolerance()` convergence criterion is satisfied when the relative change in the rotation matrix $\mathbf{T}$ from one iteration to the next is less than or equal to #. The default is `tolerance(1e-6)`.

`gtolerance(#)` is one of three criteria for declaring convergence and is rarely used. The `gtolerance()` convergence criterion is satisfied when the Frobenius norm of the gradient of the criterion function $c(\,)$ projected on the manifold of orthogonal matrices or of normal matrices is less than or equal to #. The default is `gtolerance(1e-6)`.

`ltolerance(#)` is one of three criteria for declaring convergence and is rarely used. The `ltolerance()` convergence criterion is satisfied when the relative change in the minimization criterion $c(\,)$ from one iteration to the next is less than or equal to #. The default is `ltolerance(1e-6)`.

`protect(#)` requests that # optimizations with random starting values be performed and that the best of the solutions be reported. The output also indicates whether all starting values converged to the same solution. When specified with a large number, such as `protect(50)`, this provides reasonable assurance that the solution found is the global maximum and not just a local maximum. If `trace` is also specified, the rotation matrix and rotation criterion value of each optimization will be reported.

`maxstep(#)` is a rarely used option; it specifies the maximum number of step-size halvings. The default is `maxstep(20)`.

`init(`*matname*`)` is a rarely used option; it specifies the initial rotation matrix. *matname* should be square and regular (nonsingular) and have the same number of columns as the matrix *matrix_L* to be rotated. It should be orthogonal ($\mathbf{T}'\mathbf{T} = \mathbf{T}\mathbf{T}' = \mathbf{I}$) or normal ($\text{diag}(\mathbf{T}'\mathbf{T}) = \mathbf{1}$), depending on whether orthogonal or oblique rotations are performed. `init()` cannot be combined with `random`. If neither `init()` nor `random` is specified, the identity matrix is used as the initial rotation.

`random` is a rarely used option; it specifies that a random orthogonal or random normal matrix be used as the initial rotation matrix. `random` cannot be combined with `init()`. If neither `init()` nor `random` is specified, the identity matrix is used as the initial rotation.

## Rotation criteria

In the descriptions below, the matrix to be rotated is denoted as $\mathbf{A}$, $p$ denotes the number of rows of $\mathbf{A}$, and $f$ denotes the number of columns of $\mathbf{A}$ (factors or components). If $\mathbf{A}$ is a loading matrix from `factor` or `pca`, $p$ is the number of variables and $f$ is the number of factors or components.

### Criteria suitable only for orthogonal rotations

`varimax` and `vgpf` apply the orthogonal varimax rotation (Kaiser 1958). `varimax` maximizes the variance of the squared loadings within factors (columns of $\mathbf{A}$). It is equivalent to `cf(`*1/p*`)` and to `oblimin(1)`. `varimax`, the most popular rotation, is implemented with a dedicated fast algorithm and ignores all *optimize_options*. Specify `vgpf` to switch to the general GPF algorithm used for the other criteria.

`quartimax` uses the quartimax criterion (Harman 1976). `quartimax` maximizes the variance of the squared loadings within the variables (rows of $\mathbf{A}$). For orthogonal rotations, `quartimax` is equivalent to `cf(0)` and to `oblimax`.

`equamax` specifies the orthogonal equamax rotation. `equamax` maximizes a weighted sum of the `varimax` and `quartimax` criteria, reflecting a concern for simple structure within variables (rows of $\mathbf{A}$) as well as within factors (columns of $\mathbf{A}$). `equamax` is equivalent to `oblimin(`*p/2*`)` and `cf(#)`, where $\# = f/(2p)$.

`parsimax` specifies the orthogonal parsimax rotation. `parsimax` is equivalent to `cf(#)`, where $\# = (f-1)/(p+f-2)$.

`entropy` applies the minimum entropy rotation criterion (Jennrich 2004).

`tandem1` specifies that the first principle of Comrey's tandem be applied. According to Comrey (1967), this principle should be used to judge which "small" factors be dropped.

`tandem2` specifies that the second principle of Comrey's tandem be applied. According to Comrey (1967), `tandem2` should be used for "polishing".

### Criteria suitable only for oblique rotations

`promax[(#)]` specifies the oblique promax rotation. The optional argument specifies the `promax` power. Not specifying the argument is equivalent to specifying `promax(3)`. Values less than 4 are recommended, but the choice is yours. Larger `promax` powers simplify the loadings (generate numbers closer to zero and one) but at the cost of additional correlation between factors. Choosing a value is a matter of trial and error, but most sources find values in excess of 4 undesirable in practice. The power must be greater than 1 but is not restricted to integers.

Promax rotation is an oblique rotation method that was developed before the "analytical methods" (based on criterion optimization) became computationally feasible. Promax rotation comprises an oblique Procrustean rotation of the original loadings $\mathbf{A}$ toward the elementwise #-power of the orthogonal varimax rotation of $\mathbf{A}$.

### Criteria suitable for orthogonal and oblique rotations

`oblimin[(#)]` specifies that the oblimin criterion with $\gamma = \#$ be used. When restricted to orthogonal transformations, the `oblimin()` family is equivalent to the orthomax criterion function. Special cases of `oblimin()` include

| $\gamma$ | Special case |
|---|---|
| 0 | quartimax / quartimin |
| $1/2$ | biquartimax / biquartimin |
| 1 | varimax / covarimin |
| $p/2$ | equamax |

$p$ = number of rows of $\mathbf{A}$.

$\gamma$ defaults to zero. Jennrich (1979) recommends $\gamma \leq 0$ for oblique rotations. For $\gamma > 0$, it is possible that optimal oblique rotations do not exist; the iterative procedure used to compute the solution will wander off to a degenerate solution.

`cf(#)` specifies that a criterion from the Crawford–Ferguson (1970) family be used with $\kappa = \#$. `cf(`$\kappa$`)` can be seen as $(1-\kappa)\text{cf}_1(\mathbf{A}) + (\kappa)\text{cf}_2(\mathbf{A})$, where $\text{cf}_1(\mathbf{A})$ is a measure of row parsimony and $\text{cf}_2(\mathbf{A})$ is a measure of column parsimony. $\text{cf}_1(\mathbf{A})$ attains its greatest lower bound when no row of $\mathbf{A}$ has more than one nonzero element, whereas $\text{cf}_2(\mathbf{A})$ reaches zero if no column of $\mathbf{A}$ has more than one nonzero element.

For orthogonal rotations, the Crawford–Ferguson family is equivalent to the `oblimin()` family. For orthogonal rotations, special cases include the following:

| $\kappa$ | Special case |
|---|---|
| 0 | quartimax / quartimin |
| $1/p$ | varimax / covarimin |
| $f/(2p)$ | equamax |
| $(f-1)/(p+f-2)$ | parsimax |
| 1 | factor parsimony |

$p$ = number of rows of $\mathbf{A}$.
$f$ = number of columns of $\mathbf{A}$.

`bentler` specifies that the "invariant pattern simplicity" criterion (Bentler 1977) be used.

`oblimax` specifies the oblimax criterion, which maximizes the number of high and low loadings. `oblimax` is equivalent to `quartimax` for orthogonal rotations.

`quartimin` specifies that the quartimin criterion be used. For orthogonal rotations, `quartimin` is equivalent to `quartimax`.

`target(`*Tg*`)` specifies that $\mathbf{A}$ be rotated as near as possible to the conformable matrix *Tg*. Nearness is expressed by the Frobenius matrix norm.

`partial(`*Tg W*`)` specifies that $\mathbf{A}$ be rotated as near as possible to the conformable matrix *Tg*. Nearness is expressed by a weighted (by *W*) Frobenius matrix norm. *W* should be nonnegative and usually is zero–one valued, with ones identifying the target values to be reproduced as closely as possible by the factor loadings, whereas zeros identify loadings to remain unrestricted.

## Remarks and examples

stata.com

Remarks are presented under the following headings:



### Introduction

For an introduction to rotation, see Harman (1976) and Gorsuch (1983).

`rotatemat` applies a linear transformation $\mathbf{T}$ to the specified matrix *matrix_L*, which we will call $\mathbf{A}$, so that the result $c(\mathbf{A}(\mathbf{T}')^{-1})$ minimizes a criterion function $c(\,)$ over all matrices $\mathbf{T}$ in a class of feasible transformations.

Two classes are supported: orthogonal (orthonormal) and oblique. Orthonormal rotations comprise all orthonormal matrices $\mathbf{T}$, such that $\mathbf{T}'\mathbf{T} = \mathbf{T}\mathbf{T}' = \mathbf{I}$; here $\mathbf{A}(\mathbf{T}')^{-1}$ simplifies to $\mathbf{AT}$. Oblique rotations are characterized by $\text{diag}(\mathbf{T}'\mathbf{T}) = \mathbf{1}$. All supported rotation criteria are invariant with respect to permutations of the columns and change of signs of the columns. `rotatemat` returns the solution with positive column sums and with columns sorted by the L2 norm; columns are ordered with respect to the L1 norm if the columns have the same L2 norm.

A wide variety of criteria $c(\,)$ is available, representing different ways to measure the "simplicity" of a matrix. Most of these criteria can be applied with both orthogonal and oblique rotations. A discussion of the different criteria and of the rotations with which they may be combined is provided in *Rotation criteria*.

A factor analysis of 24 psychological tests on 145 seventh- and eighth-grade school children with four retained factors is used for illustration. Factors were extracted with maximum likelihood. The loadings are reported by Harman (1976). We enter the factor loadings as a Stata matrix with 24 rows and four columns. For more information, we add full descriptive labels as comments and short labels as row names.

```
. matrix input L = (
>    601   019   388   221 \              Visual perception
>    372  -025   252   132 \              Cubes
>    413  -117   388   144 \              Paper form board
>    487  -100   254   192 \              Flags
>    691  -304  -279   035 \              General information
>    690  -409  -200  -076 \              Paragraph comprehension
>    677  -409  -292   084 \              Sentence completion
>    674  -189  -099   122 \              Word classification
>    697  -454  -212  -080 \              Word meaning
>    476   534  -486   092 \              Addition
>    558   332  -142  -090 \              Code
>    472   508  -139   256 \              Counting dots
>    602   244   028   295 \              Straight-curved capitals
>    423   058   015  -415 \              Word recognition
>    394   089   097  -362 \              Number recognition
>    510   095   347  -249 \              Figure recognition
>    466   197  -004  -381 \              Object-number
>    515   312   152  -147 \              Number-figure
>    443   089   109  -150 \              Figure-word
>    614  -118   126  -038 \              Deduction
>    589   227   057   123 \              Numerical puzzles
>    608  -107   127  -038 \              Problem reasoning
>    687  -044   138   098 \              Series completion
>    651   177  -212  -017 )              Arithmetic problems

. matrix colnames L = F1 F2 F3 F4

. matrix rownames L = visual    cubes     board
>                     flags     general   paragraph
>                     sentence  wordclas  wordmean
>                     add       code      dots
>                     capitals  wordrec   numbrec
>                     figrec    obj-num   num-fig
>                     fig-word  deduct    numpuzz
>                     reason    series    arith

. matrix L = L/1000
```

Thus using `rotatemat`, we can study various rotations of `L` without access to the full data or the correlation matrix.

## Orthogonal rotations

We can rotate the matrix L according to an extensive list of criteria, including orthogonal rotations.

### ▷ Example 1: Orthogonal varimax rotation

The default rotation, orthogonal varimax, is probably the most popular method:

```
. rotatemat L, format(%6.3f)
Rotation of L[24,4]
    Criterion               varimax
    Rotation class          orthogonal
    Kaiser normalization    off
Rotated factors
```

| | F1 | F2 | F3 | F4 |
|---|---|---|---|---|
| visual | 0.247 | 0.151 | 0.679 | 0.128 |
| cubes | 0.171 | 0.060 | 0.425 | 0.078 |
| board | 0.206 | -0.049 | 0.549 | 0.097 |
| flags | 0.295 | 0.068 | 0.504 | 0.050 |
| general | 0.765 | 0.214 | 0.117 | 0.067 |
| paragraph | 0.802 | 0.074 | 0.122 | 0.160 |
| sentence | 0.826 | 0.148 | 0.117 | -0.008 |
| wordclas | 0.612 | 0.230 | 0.290 | 0.061 |
| wordmean | 0.840 | 0.049 | 0.112 | 0.152 |
| add | 0.166 | 0.846 | -0.076 | 0.082 |
| code | 0.222 | 0.533 | 0.134 | 0.313 |
| dots | 0.048 | 0.705 | 0.257 | 0.025 |
| capitals | 0.240 | 0.500 | 0.450 | 0.020 |
| wordrec | 0.249 | 0.124 | 0.032 | 0.526 |
| numbrec | 0.178 | 0.109 | 0.106 | 0.499 |
| figrec | 0.158 | 0.076 | 0.401 | 0.510 |
| obj-num | 0.197 | 0.262 | 0.060 | 0.539 |
| num-fig | 0.096 | 0.352 | 0.311 | 0.422 |
| fig-word | 0.204 | 0.175 | 0.232 | 0.336 |
| deduct | 0.443 | 0.115 | 0.365 | 0.255 |
| numpuzz | 0.233 | 0.428 | 0.389 | 0.169 |
| reason | 0.432 | 0.120 | 0.363 | 0.256 |
| series | 0.440 | 0.228 | 0.472 | 0.184 |
| arith | 0.409 | 0.509 | 0.150 | 0.228 |

Orthogonal rotation

| | F1 | F2 | F3 | F4 |
|---|---|---|---|---|
| F1 | 0.677 | 0.438 | 0.475 | 0.352 |
| F2 | -0.632 | 0.737 | 0.049 | 0.232 |
| F3 | -0.376 | -0.458 | 0.760 | 0.268 |
| F4 | -0.011 | 0.234 | 0.441 | -0.866 |

◁

The varimax rotation $\mathbf{T}$ of $\mathbf{A}$ maximizes the (raw) varimax criterion over all orthogonal $\mathbf{T}$, which for $p \times f$ matrices is defined as (Harman 1976)

$$c_{\text{varimax}}(\mathbf{A}) = \frac{1}{p}\sum_{j=1}^{f}\left\{\left(\sum_{i=1}^{p} A_{ij}^4\right) - \frac{1}{p}\left(\sum_{i=1}^{p} A_{ij}^2\right)^2\right\}$$

The criterion $c_{\text{varimax}}(\mathbf{A})$ can be interpreted as the sum over the columns of the variances of the squares of the loadings $A_{ij}$. A column with large variance will typically consist of many small values and a few large values. Achieving such "simple" columnwise distributions is often helpful for interpretation.

## ❑ Technical note

The raw varimax criterion as defined here has been criticized because it weights variables by the size of their loadings, that is, by their communalities. This is often not desirable. A common rotation strategy is to weight all rows equally by rescaling to the same rowwise sum of squared loadings. This is known as the Kaiser normalization. You may request this normalized solution with the `normalize` option. The default in `rotatemat` and in `rotate` (see [MV] **rotate**) is not to normalize.

❑

Many other criteria for the rotation of matrices have been proposed and studied in the literature. Most of these criteria can be stated in terms of a "simplicity function". For instance, quartimax rotation (Carroll 1953) seeks to achieve interpretation within rows—in a factor analytic setup, this means that variables should have a high loading on a few factors and a low loading on the other factors. The quartimax criterion is defined as (Harman 1976)

$$c_{\text{quartimax}}(\mathbf{A}) = \left(\frac{1}{pf}\sum_{i=1}^{p}\sum_{j=1}^{f} A_{ij}^4\right) - \left(\frac{1}{pf}\sum_{i=1}^{p}\sum_{j=1}^{f} A_{ij}^2\right)^2$$

## ▷ Example 2: Orthogonal quartimax rotation

We display the quartimax solution, use blanks to represent loadings with absolute values smaller than 0.3, and suppress the display of the rotation matrix.

```
. rotatemat L, quartimax format(%6.3f) norotation blanks(0.3)
Rotation of L[24,4]
    Criterion               quartimax
    Rotation class          orthogonal
    Kaiser normalization    off
    Criterion value         -1.032898
    Number of iterations    35
Rotated factors (blanks represent abs()<.3)
```

| | F1 | F2 | F3 | F4 |
|---|---|---|---|---|
| visual | 0.374 | | 0.630 | |
| cubes | | | 0.393 | |
| board | | | 0.513 | |
| flags | 0.379 | | 0.450 | |
| general | 0.791 | | | |
| paragraph | 0.827 | | | |
| sentence | 0.838 | | | |
| wordclas | 0.669 | | | |
| wordmean | 0.860 | | | |
| add | | 0.829 | | |
| code | 0.316 | 0.521 | | |
| dots | | 0.701 | | |
| capitals | 0.348 | 0.482 | 0.393 | |
| wordrec | 0.316 | | | 0.492 |
| numbrec | | | | 0.469 |
| figrec | | | 0.382 | 0.470 |
| obj-num | | | | 0.503 |
| num-fig | | 0.357 | | 0.383 |
| fig-word | | | | |
| deduct | 0.528 | | | |
| numpuzz | 0.342 | 0.414 | 0.340 | |
| reason | 0.517 | | | |
| series | 0.543 | | 0.395 | |
| arith | 0.490 | 0.478 | | |

◁

Some of the criteria supported by `rotatemat` are defined as one-parameter families. The oblimin($\gamma$) criterion and the Crawford and Ferguson cf($\kappa$) criterion families contain the varimax and quartimax criteria as special cases; that is, they can be obtained by certain values of $\gamma$ and $\kappa$, respectively. Intermediate parameter values provide compromises between varimax's aim of column simplification and quartimax's aim of row simplification. Varimax and quartimax are equivalent to oblimin(1) and oblimin(0), respectively. A compromise, oblimin(0.5), is also known as biquartimax.

## ▷ Example 3: Orthogonal biquartimax rotation

Because the varimax and quartimax solutions are so close for our matrix `L`, the biquartimax compromise will also be rather close.

```
. rotatemat L, oblimin(0.5) format(%6.3f) norotation
  (output omitted)
```

◁

❑ Technical note

You may have noticed a difference between the output of `rotatemat` in the default case or equivalently when we type

```
. rotatemat L, varimax
```

and in other cases. In the default case, no mention is made of the criterion value and the number of iterations. `rotatemat` uses a fast special algorithm for this most common case, whereas for other rotations it uses a general gradient projection algorithm (GPF) proposed by Jennrich (2001, 2002); see also Bernaards and Jennrich (2005). The general algorithm is used to obtain the varimax rotation if you specify the option `vgpf` rather than `varimax`.

❑

The rotations we have illustrated are orthogonal—the lengths of the rows and the angles between the rows are not affected by the rotations. We may verify—we do not show this in the manual to conserve paper—that after an orthogonal rotation of `L`

```
. matlist L*L'
```

and

```
. matlist r(AT)*r(AT)'
```

return the same 24 by 24 matrix, whereas

```
. matlist r(T)*r(T)'
```

and

```
. matlist r(T)'*r(T)
```

both return a $2 \times 2$ identity matrix. `rotatemat` returns in `r(AT)` the rotated matrix and in `r(T)` the rotation matrix.

## Oblique rotations

`rotatemat` provides a second class of rotations: oblique rotations. These rotations maintain the norms of the rows of the matrix but not their inner products. In geometric terms, interpreting the rows of the matrix to be rotated as vectors, both the orthogonal and the oblique rotations maintain the lengths of the vectors. Under orthogonal transformations, the angles between the vectors are also left unchanged—these transformations comprise true reorientations in space and reflections. Oblique rotations do not conserve angles between vectors. If the vectors are orthogonal before rotations—as will be the case if we are rotating factor or component loading matrices—this will no longer be the case after the rotation. The "freedom" to select angles between the rows allows oblique rotations to generate simpler loading structures than the orthogonal rotations—sometimes much simpler. In a factor analytic setting, the disadvantage is, however, that the rotated factors are correlated.

`rotatemat` can obtain oblique rotations for most of the criteria that are available for orthogonal rotations; some of the criteria (such as the entropy criterion) are available only for the orthogonal case.

## Example 4: Oblique oblimin rotation

We illustrate with the psychological tests matrix L and apply the oblique oblimin criterion.

```
. rotatemat L, oblimin oblique format(%6.3f) blanks(0.3)
Rotation of L[24,4]
    Criterion               oblimin(0)
    Rotation class          oblique
    Kaiser normalization    off
    Criterion value         .1957363
    Number of iterations    78
Rotated factors (blanks represent abs()<.3)
```

| | F1 | F2 | F3 | F4 |
|---|---|---|---|---|
| visual | | 0.686 | | |
| cubes | | 0.430 | | |
| board | | 0.564 | | |
| flags | | 0.507 | | |
| general | 0.771 | | | |
| paragraph | 0.808 | | | |
| sentence | 0.865 | | | |
| wordclas | 0.560 | | | |
| wordmean | 0.857 | | | |
| add | | | 0.864 | |
| code | | | 0.460 | 0.305 |
| dots | | | 0.701 | |
| capitals | | 0.437 | 0.442 | |
| wordrec | | | | 0.571 |
| numbrec | | | | 0.543 |
| figrec | | 0.314 | | 0.540 |
| obj-num | | | | 0.584 |
| num-fig | | | | 0.438 |
| fig-word | | | | 0.341 |
| deduct | 0.325 | | | |
| numpuzz | | 0.344 | 0.347 | |
| reason | 0.311 | | | |
| series | | 0.417 | | |
| arith | | | 0.428 | |

```
Oblique rotation
```

| | F1 | F2 | F3 | F4 |
|---|---|---|---|---|
| F1 | 0.823 | 0.715 | 0.584 | 0.699 |
| F2 | -0.483 | 0.019 | 0.651 | 0.213 |
| F3 | -0.299 | 0.587 | -0.435 | 0.207 |
| F4 | -0.006 | 0.379 | 0.213 | -0.651 |

The option `oblique` requested an oblique rotation rather than the default `orthogonal`. You may verify that `r(AT)` equals `L * inv(r(T)')` within reasonable roundoff with

```
. matlist r(AT) - L * inv(r(T)')
```
(*output omitted*)

The correlation between the rotated dimensions is easily obtained.

```
. matlist r(T)' * r(T)
```

| | F1 | F2 | F3 | F4 |
|---|---|---|---|---|
| F1 | 1 | | | |
| F2 | .4026978 | 1 | | |
| F3 | .294928 | .2555824 | 1 | |
| F4 | .4146879 | .3784689 | .3183115 | 1 |

## Promax rotation

`rotatemat` also offers promax rotation.

### ▷ Example 5: Oblique promax rotation

We use the matrix `L` to illustrate promax rotation.

```
. rotatemat L, promax blanks(0.3) format(%6.3f)
Rotation of L[24,4]
    Criterion               promax(3)
    Rotation class          oblique
    Kaiser normalization    off
Rotated factors (blanks represent abs()<.3)
```

| | F1 | F2 | F3 | F4 |
|---|---|---|---|---|
| visual | | 0.775 | | |
| cubes | | 0.487 | | |
| board | | 0.647 | | |
| flags | | 0.572 | | |
| general | 0.786 | | | |
| paragraph | 0.825 | | | |
| sentence | 0.888 | | | |
| wordclas | 0.543 | | | |
| wordmean | 0.878 | | | |
| add | | | 0.921 | |
| code | | | 0.466 | |
| dots | | | 0.728 | |
| capitals | | 0.468 | 0.441 | |
| wordrec | | | | 0.606 |
| numbrec | | | | 0.570 |
| figrec | | 0.364 | | 0.539 |
| obj-num | | | | 0.610 |
| num-fig | | | | 0.425 |
| fig-word | | | | 0.337 |
| deduct | | 0.323 | | |
| numpuzz | | 0.369 | 0.336 | |
| reason | | 0.322 | | |
| series | | 0.462 | | |
| arith | | | 0.436 | |

```
Oblique rotation

------------------------------------------------
             |      F1       F2       F3       F4
-------------+----------------------------------
          F1 |   0.841    0.829    0.663    0.743
          F2 |  -0.462    0.020    0.614    0.215
          F3 |  -0.282    0.478   -0.386    0.159
          F4 |  -0.012    0.290    0.184   -0.614
------------------------------------------------
```

The correlation between the rotated dimensions can be obtained as

```
. matlist r(T)' * r(T)

             |      F1         F2         F3         F4
-------------+----------------------------------------
          F1 |       1
          F2 |  .5491588          1
          F3 |  .3807942  .4302401          1
          F4 |  .4877064  .5178414  .4505817          1
```

## Stored results

`rotatemat` stores the following in `r()`:

Scalars

| | |
|---|---|
| `r(f)` | criterion value |
| `r(iter)` | number of GPF iterations |
| `r(rc)` | return code |
| `r(nnconv)` | number of nonconvergent trials; `protect()` only |

Macros

| | |
|---|---|
| `r(cmd)` | `rotatemat` |
| `r(ctitle)` | descriptive label of rotation method |
| `r(ctitle12)` | version of `r(ctitle)` at most 12 characters long |
| `r(criterion)` | criterion name (e.g., `oblimin`) |
| `r(class)` | `orthogonal` or `oblique` |
| `r(normalization)` | `kaiser` or `none` |
| `r(carg)` | criterion argument |

Matrices

| | |
|---|---|
| `r(T)` | optimal transformation $\mathbf{T}$ |
| `r(AT)` | optimal $\mathbf{AT} = \mathbf{A}(\mathbf{T}')^{-1}$ |
| `r(fmin)` | minimums found; `protect()` only |

## Methods and formulas

`rotatemat` minimizes a scalar-valued criterion function $c(\mathbf{AT})$ with respect to the set of orthogonal matrices $\mathbf{T}'\mathbf{T} = \mathbf{I}$, or $c(\mathbf{A}(\mathbf{T}')^{-1})$ with respect to the normal matrix, $\text{diag}(\mathbf{T}'\mathbf{T}) = \mathbf{1}$. For orthonormal $\mathbf{T}$, $\mathbf{T} = (\mathbf{T}')^{-1}$.

The rotation criteria can be conveniently written in terms of scalar-valued functions; see Bernaards and Jennrich (2005). Define the inner product $\langle \mathbf{A}, \mathbf{B} \rangle = \text{trace}(\mathbf{A}'\mathbf{B})$. $|\mathbf{A}| = \sqrt{\langle \mathbf{A}, \mathbf{A} \rangle}$ is called the Frobenius norm of the matrix $\mathbf{A}$. Let $\mathbf{\Lambda}$ be a $p \times k$ matrix. Denote by $\mathbf{X}^2$ the direct product $\mathbf{X} \cdot \mathbf{X}$. See Harman (1976) for information on many of the rotation criteria and references to the authors originally proposing the criteria. Sometimes we list an alternative reference. Our notation is similar to that of Bernaards and Jennrich (2005).

`rotatemat` uses the iterative "gradient projection algorithm" (Jennrich 2001, 2002) for the optimization of the criterion over the permissible transformations. Different versions are provided for optimal orthogonal and oblique rotations; see Bernaards and Jennrich (2005).

### Varimax (orthogonal only)

Varimax is equivalent to oblimin with $\gamma = 1$ or to the Crawford–Ferguson family with $\kappa = 1/p$; see below.

### Quartimax (orthogonal only)

$$c(\mathbf{\Lambda}) = \sum_i \sum_r \lambda_{ir}^4 = -\frac{1}{4} \left\langle \mathbf{\Lambda}^2, \mathbf{\Lambda}^2 \right\rangle$$

### Equamax (orthogonal only)

Equamax is equivalent to oblimin with $\gamma = p/2$ or to the Crawford–Ferguson family with $\kappa = f/(2p)$; see below.

### Parsimax (orthogonal only)

Parsimax is equivalent to the Crawford–Ferguson family with $\kappa = (f-1)/(p+f-2)$; see below.

### Entropy (orthogonal only); see Jennrich (2004)

$$c(\mathbf{\Lambda}) = -\frac{1}{2} \left\langle \mathbf{\Lambda}^2, \log \mathbf{\Lambda}^2 \right\rangle$$

### Tandem principal 1 (orthogonal only); see Comrey (1967)

$$c(\mathbf{\Lambda}) = -\left\langle \mathbf{\Lambda}^2, (\mathbf{\Lambda}\mathbf{\Lambda}')^2 \mathbf{\Lambda}^2 \right\rangle$$

### Tandem principal 2 (orthogonal only); see Comrey (1967)

$$c(\mathbf{\Lambda}) = \left\langle \mathbf{\Lambda}^2, \{\mathbf{1}\mathbf{1}' - (\mathbf{\Lambda}\mathbf{\Lambda}')^2\} \mathbf{\Lambda}^2 \right\rangle$$

### Promax (oblique only)

Promax does not fit in the maximizing-of-a-simplicity-criterion framework that is at the core of `rotatemat`. The promax method (Hendrickson and White 1964) was proposed before computing power became widely available. The promax rotation comprises three steps:

1. Perform an orthogonal rotation on $\mathbf{A}$; `rotatemat` uses varimax.
2. Raise the elements of the rotated matrix to some power, preserving the signs of the elements. Typically, the power is taken from the range [2,4]. This operation is meant to distinguish more clearly between small and large values.
3. The matrix from step 2 is used as the target for an oblique Procrustean rotation from the original matrix $\mathbf{A}$. The method to compute this rotation in promax is different from the method in the `procrustes` command (see [MV] **procrustes**). The latter produces the real least-squares oblique rotation; promax uses an approximation.

### Oblimin; see Jennrich (1979)

$$c(\mathbf{\Lambda}) = \frac{1}{4}\left\langle \mathbf{\Lambda}^2, \{\mathbf{I} - (\gamma/p)\mathbf{1}\mathbf{1}'\}\mathbf{\Lambda}^2(\mathbf{1}\mathbf{1}' - \mathbf{I})\right\rangle$$

Orthomax and oblimin are equivalent when restricted to orthogonal rotations. Special cases of `oblimin()` include the following:

| $\gamma$ | Special case |
|---|---|
| 0 | quartimin |
| $1/2$ | biquartimin |
| $p/2$ | equamax |
| 1 | varimax |

### Crawford and Ferguson (1970) family

$$c(\mathbf{\Lambda}) = \frac{1-\kappa}{4}\left\langle \mathbf{\Lambda}^2, \mathbf{\Lambda}^2(\mathbf{1}\mathbf{1}' - \mathbf{I})\right\rangle + \frac{\kappa}{4}\left\langle \mathbf{\Lambda}^2, (\mathbf{1}\mathbf{1}' - \mathbf{I})\mathbf{\Lambda}^2\right\rangle$$

When restricted to orthogonal transformations, `cf()` and `oblimin()` are in fact equivalent. Special cases of `cf()` include the following:

| $\kappa$ | Special case |
|---|---|
| 0 | quartimax |
| $1/p$ | varimax |
| $f/(2p)$ | equamax |
| $(f-1)/(p+f-2)$ | parsimax |
| 1 | factor parsimony |

### Bentler's invariant pattern simplicity; see Bentler (1977)

$$c(\mathbf{\Lambda}) = \log[\det\{(\mathbf{\Lambda}^2)'\mathbf{\Lambda}^2\}] - \log(\det[\mathrm{diag}\{(\mathbf{\Lambda}^2)'\mathbf{\Lambda}^2\}])$$

### Oblimax

$$c(\mathbf{\Lambda}) = -\log(\langle \mathbf{\Lambda}^2, \mathbf{\Lambda}^2 \rangle) + 2\log(\langle \mathbf{\Lambda}, \mathbf{\Lambda} \rangle)$$

For orthogonal transformations, oblimax is equivalent to quartimax; see above.

### Quartimin

$$c(\mathbf{\Lambda}) = \sum_{r \neq s} \sum_{i} \lambda_{ir}^2 \lambda_{is}^2 = -\frac{1}{4} \langle \mathbf{\Lambda}^2, \mathbf{\Lambda}^2(\mathbf{1}\mathbf{1}' - \mathbf{I}) \rangle$$

### Target

$$c(\mathbf{\Lambda}) = \frac{1}{2}|\mathbf{\Lambda} - \mathbf{H}|^2$$

for given target matrix $\mathbf{H}$.

### Partially specified target

$$c(\mathbf{\Lambda}) = |\mathbf{W} \cdot (\mathbf{\Lambda} - \mathbf{H})|^2$$

for given target matrix $\mathbf{H}$, nonnegative weighting matrix $\mathbf{W}$ (usually zero–one valued) and with $\cdot$ denoting the direct product.

## References


Bentler, P. M. 1977. Factor simplicity index and transformations. *Psychometrika* 42: 277–295.

Bernaards, C. A., and R. I. Jennrich. 2005. Gradient projection algorithms and software for arbitrary rotation criteria in factor analysis. *Educational and Psychological Measurement* 65: 676–696.

Carroll, J. B. 1953. An analytical solution for approximating simple structure in factor analysis. *Psychometrika* 18: 23–38.

Comrey, A. L. 1967. Tandem criteria for analytic rotation in factor analysis. *Psychometrika* 32: 277–295.

Crawford, C. B., and G. A. Ferguson. 1970. A general rotation criterion and its use in orthogonal rotation. *Psychometrika* 35: 321–332.

Gorsuch, R. L. 1983. *Factor Analysis*. 2nd ed. Hillsdale, NJ: Lawrence Erlbaum.

Harman, H. H. 1976. *Modern Factor Analysis*. 3rd ed. Chicago: University of Chicago Press.

Hendrickson, A. E., and P. O. White. 1964. Promax: A quick method for rotation to oblique simple structure. *British Journal of Statistical Psychology* 17: 65–70.

Horst, P. 1965. *Factor Analysis of Data Matrices*. New York: Holt, Rinehart & Winston.

Jennrich, R. I. 1979. Admissible values of $\gamma$ in direct oblimin rotation. *Psychometrika* 44: 173–177.

——. 2001. A simple general procedure for orthogonal rotation. *Psychometrika* 66: 289–306.

——. 2002. A simple general method for oblique rotation. *Psychometrika* 67: 7–20.

——. 2004. Rotation to simple loadings using component loss functions: The orthogonal case. *Psychometrika* 69: 257–273.

Kaiser, H. F. 1958. The varimax criterion for analytic rotation in factor analysis. *Psychometrika* 23: 187–200.

## Also see

[MV] **rotate** — Orthogonal and oblique rotations after factor and pca

[MV] **procrustes** — Procrustes transformation