CMAT© Newsletter: July 2018

Wolfgang M. Hartmann

July 2018

Contents

1 General Remarks

Development on CMAT is no longer as intensive as in earlier years since I am now working on two other projects. One of those at least applies CMAT and therefore I'm still adding features to CMAT. The `mds()` function

< gof,conf,resi > = mds("unis",data,optn<,...> . >)	UNISCALE
< gof,conf,resi,grad > = mds("kyst",data,optn<,...> . >)	KYST
< gof,conf,resi,wgt,eval,evec > = mds("cosp",data,optn<,...> . >)	COSPA
< gof,conf,resi,wgt,eval,evec > = mds("inds",data,optn<,...> . >)	INDSCAL
< gof,conf,resi,wgt,eval,evec > = mds("sums",data,optn<,...> . >)	SUMSCAL
< gof,conf,resi,wgt,eval,evec > = mds("alsc",data,optn<,...> . >)	ALSCAL
< gof,conf,resi,wgt,eval,evec > = mds("torg",data,optn<,...> . >)	TORGerson
< gof,conf,resi,wgt,cov > = mds("mult",data,optn<,...> . >)	MULTISCALE
< gof,conf,resi,wgt,cov > = mds("smac",data,optn<,...> . >)	SMACOF
< gof,conf,resi,wgt,cov > = mds("maxl",data,optn<,...> . >)	MAXLIK

was worked over and two new algorithms were added,

- a simple metric version of the ALSCAL method (de Leeuw, 2012; Takane, Young, & de Leeuw, 1977),
- the metric SMACOF algorithm by de Leeuw (2005), Mair, de Leeuw, Borg, & Groenen (2016),
- the metric MAXLIK algorithm, which is a version of MULTISCALE developed by Ramsay (1977).

In addition, the return arguments have been worked over.

1. When the `"ndim"` or the `"dimmin"` and `"dimmax"` input options specify only one dimension d, the return arguments are now:

   "gof" is a column vector,

   "conf" either matrix or tensor is returned:

   (a) no `"iase"` option: no standard errors: matrix `xm[n,d]` is returned

   (b) with `"iase"` option: with standard errors: tensor `xt[4,n,d]` is returned with estimates, ASEs, LCIs and UCIs in its slides.

   "resi" tensor is returned:

   (a) for only $m = 1$ input matrix: tensor resi[4,n,n] of residuals is returned, where the first slide corresponds to the input data, the second slide to the preprocessed (transformed) input data, the

third slide the fitted distances and the last slide to the residuals (difference between slides 2 and 3)

(b) for $m > 1$: input matrices: tensor resi[m,4,nobs,nobs] is returned

"wgt" for TORGerson, ALSCAL, COSPA, INDSCAL, SUMSCAL methods

TORGerson, ALSCAL : missing values are returned

MULTISCALE : matrix or tensor:

(a) for $m = 1$ input matrix: either diagonal or symmetric $d \times d$ matrix

(b) for $m > 1$ input matrices: tensor $m \times d \times d$]

COSPA, INDSCAL, SUMSCAL : matrix or tensor:

(a) for $m = 1$ input matrix: diagonal $d \times d$ matrix

(b) for $m > 1$ input matrices: tensor $m \times d \times d$]

KYST : $n \times d$ matrix of gradients

"grad" for KYST: returns gradient as a $n \times d$ matrix

"eval" for TORGerson, ALSCal, COSPa, INDScal, SUMScal only returns vector of d values (or scalar for d=1)

"evec" for TORGerson, ALSCal, COSPa, INDScal, SUMScal only returns $n \times d$ matrix of eigen vectors

"cov" for MULTiscal and SMACof only: returns tensor [n,d,d] of n symmetric $d \times d$ covariance matrices of the $n \times d$ **X** configuration, one for each of the n points

2. Whenever the `"dimmin"` and `"dimmax"` input options specify more than one dimensions, $nd > 1$, i.e. $d_1, \ldots, d_v$, then

   - the `"gof` return argument becomes a matrix with v columns,
   - and all other return arguments become lists or tensors of v data objects.

1.1 New Functions

mdu() for multidimensional unfolding (the rectangular version of `mds()`),

xplot() for plotting a onedimensional configuration of points `xplot()`),

1.2 Fixed Bugs

2 Modifications of Features

2.1 Modifications of `anacor()` and `anaprof()`

The names of the input options for specifying the minimum and maximum number of dimensions were changed from `dim1` and `dim2` into now `dimmin` and `dimmax`.

3 Extensions to the Language

4 Extensions to Various Functions

4.1 Extension of `mds()` Function

< gof,conf,resi,... > = mds("meth",data,optn<,xini<,wini <,tra<,ceq<,ref<,pspl<,sew> .. >)

< gof,conf,resi > = mds("unis",data,optn<,...> . >)	UNISCALE
< gof,conf,resi,grad > = mds("kyst",data,optn<,...> . >)	KYST
< gof,conf,resi,wgt,cov > = mds("mult",data,optn<,...> . >)	MULTISCALE
< gof,conf,resi,wgt,cov > = mds("smac",data,optn<,...> . >)	SMACOF
< gof,conf,resi,wgt,cov > = mds("maxl",data,optn<,...> . >)	MAXLIK
< gof,conf,resi,wgt,eval,evec > = mds("alsc",data,optn<,...> . >)	ALSCAL
< gof,conf,resi,wgt,eval,evec > = mds("torg",data,optn<,...> . >)	TORGerson
< gof,conf,resi,wgt,eval,evec > = mds("cosp",data,optn<,...> . >)	COSPA
< gof,conf,resi,wgt,eval,evec > = mds("inds",data,optn<,...> . >)	INDSCAL
< gof,conf,resi,wgt,eval,evec > = mds("sums",data,optn<,...> . >)	SUMSCAL

Purpose: This function implements a variety of *metric* and *nonmetric* methods for 2-way and 3-way multidimensional scaling. Similar to cluster analysis, multidimensional scaling analyzes the relations between the rows of a data set (observations, cases) and is therefore different of most statistical methods which analyze the relations between the columns (variables, items) of a data set.

All methods analyze symmetric (dis)similarity (distance) matrices $\mathbf{S} = (s_{ij})$. Only KYST and MULTISCAL are able to analyze unsymmetric but square data matrices. When the input consists of rectangular raw data matrices, the data is transformed automatically into symmetric distance matrices. A similar analysis but of rectangular (dis)similarity (distance) matrices is possible via multidimensional unfolding (MDU).

The result is usually a $n \times d$ configuration matrix $\hat{\mathbf{X}}$ of n points in a small $d << p$ dimensional space, whose interpoint distances $d(x_i, x_j)$ fit in some way the distances among the larger p dimensional space of the input data $\mathbf{S}$. If $d = 2$ or $d = 3$, the plot of the $n \times d$ result configration may uncover the most important relations among the observations of the data, such as clusters of similar subjects.

If the data is a tensor consisting of m data matrices belonging to different subjects or treatments different models for the model distance matrix $\mathbf{D}_i$,

$$X_i = X * W_i \quad , \text{ for } i = 1, \ldots, m$$

can be applied among them:

identity $\mathbf{W}$ is the identity $\mathbf{I}$ matrix. This model can be fitted via SMACOF, KYST, ALSCAL, MAXLIK, and MULTISCALE.

diagonal $\mathbf{W}$ is a $d \times d$ positive diagonal matrix: this model is also known as Horan's model and can be fitted via COSPA, SUMSCAL, INDSCAL, and MULTISCALE.

full (symmetric) model: $\mathbf{W}$ is a $d \times d$ symmetric matrix with a positive diagonal. This model can be fitted by MULTISCALE.

The following estimation methods are implemented for 2-way analyses:

TORGERSON Torgerson metric scaling which is commonly used for obtaining starting configurations for other, more sophisticated methods of `mds()` and `mdu()`.

ALSCAL Takane, Young, & de Leeuw (1977) metric MDS method (see de Leeuw, 2012)

KYST Kruskal-Young-Shepard-Torgerson metric and nonmetric scaling; see Young (1987), Kruskal, Young, & Seery (1978)

SMACOF metric and nonmetric method by de Leeuw (1984, 1994, 2007): these are iterative ALS methods suitable for problems with many observations

MAXLIK maximum likelihood method similar to MULTISCALE by Ramsay (1977, 1978) : this method may also compute ellipsoidal confidence intervals of the point coordinates by de Leeuws() method

MULTISCALE maximum likelihood method by Ramsay (1977, 1978): is also useful for 3-way analyses, this method may also compute asymptotic standard errors for the unrotated point coordinates using the diagonal of the Hessian matrix of the maximum likelihood vriterion (from which ellipsoidal confidence intervals of the point estimates can be drawn).

The following methods are specifically designed for metric 3-way analyses:

COSPA 3-way (metric) COmmon SPace Analysis by Schoenemann (1972), default for 3-way, but can be used for 2-way analysis

INDSCAL 3-way (metric) INdividual Differences Scaling by Carroll & Chang (1970) using the Canonical Decomposition (CANDECOMP) technique is based on Horan's (1969) model, cannot be used for 2-way analysis

SUMSCAL 3-way (metric) individual differences scaling by de Leeuw & Pruzansky (1978), normally for 3-way, but can be used for 2-way too.

The result of the methods is an $n \times d$ configuration matrix $\hat{\mathbf{X}}$ of n points in a small $d << p$ Euclidean space. For 3-way analyses of m dissimilarity matrices we also obtain the estimates of m weight matrices.

Due to the invariance of Euclidean distances the solution $\hat{\mathbf{X}}$ is normalized

TORGERSON centered and rotated toward principal components

ALSCAL centered and rotated w.r.t. `"rot"` and `"rotp"` options

KYST centered and rotated w.r.t. `"rot"` and `"rotp"` options

SMACOF centered and rotated w.r.t. `"rot"` and `"rotp"` options

MAXLIK centered and rotated w.r.t. `"rot"` and `"rotp"` options

MULTISCALE centered and rotated toward principal components

COSPA $\hat{\mathbf{X}}$ is centered and weighted to unit SSQ, and the dimensions are ordered w.r.t. to SSQ of weight matrices

INDSCAL $\hat{\mathbf{X}}$ is centered and weighted to unit SSQ, and the dimensions are ordered w.r.t. to SSQ of weight matrices

SUMSCAL $\hat{\mathbf{X}}$ is centered and weighted to unit SSQ, and the dimensions are ordered w.r.t. to SSQ of weight matrices

Input: data This function permits the following forms of data input:

1. $n \times p$ matrices or $K \times n \times p$ 3-way tables (tensors) of *raw* data.
2. symmetric $n \times n$ matrices or $K \times n \times n$ 3-way tables (tensors) of distance (dissimilarity) data.
3. symmetric $n \times n$ matrices or $K \times n \times n$ 3-way tables (tensors) of correlation, covariance, or scalar product (similarity) data.

optn This argument must be specified in form of a two column matrix where the first column defines the option as string value (in quotes) and the second column can be used for a numeric or string specification of the option. See table below for content.

xini an $n \times d$ matrix of initial values for the $\mathbf{X}$ configuration of points

Option	Second Column	Meaning
"addc"		add constant if distance data do not satisfy triangle inequality
"blockd"		KYST????
"cl"	string	
"covsing"	real	
"dimmax"	int	maximum number dimensions d (def=2)
"dimmin"	int	minimum number dimensions d (def=2)
"dimdif"	int	step size dimensions d (def=1)
"data"	string	kind of input data
	"raw"	$n \times p$ raw data
	"diss"	dissimilarities
	"simi"	similarities
	"dist"	distances
	"corr"	correlations
	"cova"	covariances
	"scp"	scalar product data
"evsing"		
"fix"		
"gutlin"		apply Guttman-Lingoes transformation
"maxit"	int	number of iterations, the default is: ALSCAL = 1000, INDSAL = KYST = SMACOF = 5000 MULTISCAL = 200, otherwise = 400
"ndim"	int	number dimensions d (def=2)

Option	Second Column	Meaning
"noplot"		do not plot
"nopr"		do not print
"pcoef"		print X and W configurations
"pdat"		print input data
"pdis"		print distance matrices
"phis"		print iteration history
"print"	int	controls printed output
"pres"		printout residuals
"plotf"	char	plotting into text file (with larger line size)
"plotco"	char	configuration plotting (dimensions pairwise)
	"non"	
	"som"	
	"all"	
"plotsc"	char	scatter plotting (data vs. model)
	"non"	
	"som"	
	"all"	
	"joi"	
"plowgt"	char	subject weights plotting (dimensions pairwise)
	"non"	
	"som"	
	"all"	
"rot"	char	orthogonal rotation method, default is VARIMAX
		must not be specified if "rotp" is specified
	"quamax"	QUARTIMAX
	"varmax"	VARIMAX
	"equmax"	EQUAMAX
	"parmax"	PARSIMAX
	"facpar"	FACTPARS
"rotp"	real	γ parameter for orthogonal rotation
		must not be specified if "rot" is specified
"seed"	int	seed of random generator (def: time of day)
"sing"	real	singularity criterion (def=1.e-8)
"xtol"	real	termination criterion (def=1.e-5)

Specific options for the INDSCAL, COSPA, and SUMSCAL methods:

This three methods are specifically designed for the 3-way metric MDS of a set of K scalar product matrices. For input raw data the `mds` function will produce distance or scalar product matrices which are suitable. They only fit 3-way scalar product data in a least squares sense and are therefore rather sensitive to outliers of the data. All other input data types, like (squared) Euclidean distances, are transformed internally to scalar product data.

Option	**Second Column**	**Meaning**
`"dist"`	string	form of distance function
	"euc"	use Euclidean distances (def)
	"squ"	use squared Euclidean distances
`"norm"`		use normed scalar products
`"plin"`	int	print only selected steps of INDSCAL iteration

Specific options for the ALSCAL method:

Option	**Second Column**	**Meaning**
`"ftol"`	real	termination test for stress value, def=.0001

Specific options for the more general KYST methods:

KYST uses more general algorithms and is applicable to a wider range of input data. KYST also permits to fit city-block distances and nonlinear or monotone regression between the data and the model distances. KYST can also deal with a reasonable amount of missing data.

Option	Second Column	Meaning
`"acsav"`	real	for termination test, def=0.66
`"cutof"`	real	threshold for removing small distances from fit
`"cosav"`	real	for termination test, def=0.66
`"coord"`	char	type of normalization
	"as-is"	no normalization
	"stand"	normalize at each iteration
	"rotat"	rotate to principal components, def
`"diag"`	char	do or don't use diagonal of data matrix
	"absent"	do not use diagonal
	"present"	use diagonal (def)
`"dpow"`	real	power of distance function, def=2. should be in [1., 4.]
`"ftol"`	real	termination test for stress value, def=.01
`"gtol"`	real	termination test for 2-norm of gradient, def=0
`"pre-it"`	int	number of iterations for start estimates
`"poldeg"`	int	polynomial degree of regression should be in $[1, 4]$
`"regres"`	char	type of regression between data and model values
	"asc"	isotone ascending (def)
	"des"	isotone descending
	"poly"	polynomial with intercept
	"poln"	polynomial without intercept
`"split"`	char	split data values in groups (for unfolding)
	"byrow"	
	"bygroup"	
	"bydeck"	
`"sfrm"`	int	kind of stress formula for optimization =1: default, =2: for unfolding
`"sratst"`	real	for termination test, def=.999
`"start"`	char	method for initial estimates
	"rand"	random
	"tors"	Torgerson-Young quasi nonmetric
`"ties"`	char	type of tie processing in isotone regression
	"prim"	type 1 processing
	"seco"	type 2 processing
`"unsym"`		input of unsymmetric data matrix (does not refer to `"data"` `"raw"` input)

Specific options for the MULTISCALE method:

Option	Second Column	Meaning
"alpha"	real	for CI of ML estimation (def=0.05)
"ccorr"		compute and print correlations of log dissimilarities between matrices
"coord"	char	type of normalization
	"as-is"	no normalization
	"rotat"	rotate to principal components (def)
"dlogn"		error distribution: lognormal
"dnorm0"		error distribution: normal with beta=0
"dnorm5"		error distribution: normal with beta=.5
"dnorm1"		error distribution: normal with beta=1
"itxmax"	int	maximum iterations for X (coordinates) optimization
"itwmax"	int	maximum iterations for W (weights) optimization
"itpmax"	int	maximum iterations for P (spline) optimization
"itamax"	int	maximum iterations for A (stand.error) optimization
"knots"	int	number (inner) knots for spline transform (is internally increased by two more boundary knots)
"mod"	char	subjective weighting model
	"ide"	identity model
	"dia"	(positive) diagonal weights
	"ful"	full $d \times d$ weight matrices
"noase"		does not compute standard errors
"orig"	int	=2: use original version of MULTISCALE program
		=0: use modified version
"pcor"		print the n $d \times d$ asymptotic correlation matrices of coordinate estimates
"pcov"		print the n $d \times d$ asymptotic covariance matrices of coordinate estimates
"plis"		print line search information
"ptab"		print some table output
"regres"	char	type of regression between data and model values
	"scale"	interval transform
	"power"	power transform
	"spline"	spline transform

Option	Second Column	Meaning
`"tech"`	char	optimization technique (see NLP)
`"vers"`	int	version of the algorithm (see below)
	0	parametrization and optimization as original version by J. Ramsay
	1	parametrization as vers=0 using `nlp()`
	2	summarize m optimizations for = "dia" or "full" weights
	3	optimization w.r.t. all model parameters
`"varobs"`	char	stimulus variance level: A fit
	"con"	constant (default)
	"obs"	stimuliwise (fit n add. parameters)
	"pair"	pairwise (set $n*(n+1)/2$ estimates)
`"varsub"`	char	subject variance level
	"con"	constant
	"subw"	subjectwise (default) set m add. estimates

The number of parameters $p = p_1+p_2+p_3$ for the MULTISCALE approach is:

1. `model` specification:

$$p_1 = \begin{cases} n*d - d*(d+1)/2 & \text{identity} \\ n*d + (m-1)*(d-1) - d & \text{diagonal} \\ n*d + (m-1)*d(d+1)/2 - m - d + 1 & \text{full} \end{cases}$$

2. `varobs` specification:

$$p_2 = \begin{cases} 0 & \text{constant} \\ n-1 & \text{stimuliwise} \\ n*(n-1)/2 - 1 & \text{pairwise} \end{cases}$$

3. `varsub` specification:

$$p_3 = \begin{cases} 0 & \text{constant} \\ m-1 & \text{subjectwise} \end{cases}$$

A maximum of four sets of parameters is to estimate:

CONF the $n*d$ coordinates of n points in d dimensions.

WGT if "mod" equal "dia" or "full": the $m*nw$ with m weight matrices and $nw = d$ for diagonal or $nw = d*(d+1)/2$ for full symmetric weight matrices

SPLINE for "regres" equal "spline" and "knots" ¿ 0: there are m sets of "knots" parameters to estimate.

VARS if "varobs" equal "stim" n more parameters are estimated using optimization. (If "varobs" equal "pair" there are $n*(n-1)/2$ more parameters estimated without optimization.)

1. For "vers" equal 0 (original version by Ramsay) and equl 1: each of the parameter (4 or less) groups is estimated separately, the m groups WGT and SPLINE by separate optimizations, i.e. each cycle consists of a max of $2 + 2m$ optimizations.
2. For "vers" equal 2 the m parameter groups of WGT and SPLINE are done together, i.e. each cycle consists of no more than 4 optimizations.
3. For "vers" equal 3 the optimization is done across all parameters.

Specific options for the MAXLIK method:

Option	Second Column	Meaning
"alpha"	real	for CI of ML estimation (def=0.05)
"coord"	char	type of normalization
	"as-is"	no normalization
	"rotat"	rotate to principal components (def)
"design"	"rams"	rank deficient design: Ramsay (1977)
		$n \times nd$ parms, $nd * (nd+1)/2$ nullities
	"full"	full rank design: estimate only part of X
		with only $n * nd - nd * (nd+1)/2$ estimates
	"mask"	with $n \times nd$ parameters,
		subject to $nd * (nd+1)/2$ Equality BCs
"dlogn"		error distribution: lognormal
"dnorm0"		error distribution: normal with beta=0
"dnorm5"		error distribution: normal with beta=.5
"dnorm1"		error distribution: normal with beta=1
"itxmax"	int	maximum iterations for X (coordinates) optimization
"itamax"	int	maximum iterations for A (stand.error) optimization
"noase"		does not compute standard errors
"orig"	int	=1: use general inverse of Hessian
		=0: use
"pcor"		print the n $d \times d$ asymptotic correlation matrices of coordinate estimates
"pcov"		print the n $d \times d$ asymptotic covariance matrices of coordinate estimates
"ptab"		print some table output
"regres"	char	type of regression between data and model values
	"scale"	interval transform
	"power"	power transform
	"spline"	spline transform

Option	Second Column	Meaning
"tech"	char	optimization technique (see NLP)
"varobs"	char	stimulus variance level: A fit
	"con"	constant (default)
	"obs"	stimuliwise (fit n add. parameters)
"varsub"	char	subject variance level
	"con"	constant
	"subw"	subjectwise (default) set m add. estimates

The number of parameters $p = p_1 + p_2 + p_3$ for the MULTISCALE approach is:

1. `model` specification:

$$p_1 = \begin{cases} n*d - d*(d+1)/2 & \text{identity} \\ n*d + (m-1)*(d-1) - d & \text{diagonal} \\ n*d + (m-1)*d(d+1)/2 - m - d + 1 & \text{full} \end{cases}$$

2. `varobs` specification:

$$p_2 = \begin{cases} 0 & \text{constant} \\ n-1 & \text{stimuliwise} \\ n*(n-1)/2 - 1 & \text{pairwise} \end{cases}$$

3. `varsub` specification:

$$p_3 = \begin{cases} 0 & \text{constant} \\ m-1 & \text{subjectwise} \end{cases}$$

A maximum of two sets of parameters is to estimate:

CONF the $n*d$ coordinates of n points in d dimensions.

VARS if "varobs" equal "stim" n more parameters are estimated using optimization. (If "varobs" equal "pair" there are $n*(n-1)/2$ more parameters estimated without optimization.)

Specific options for the SMACOF method:

Option	Second Column	Meaning
`"ftol"`	real	termination test for stress value, def=1e-6
`"plin"`	int	print only selected steps of SMACOF iteration
`"poldeg"`	int	polynomial degree of regression
		should be in $[1, 4]$ (see KYST)
`"relax"`		relaxation for convergence problems
`"ties"`	char	ties in isotone regression (see KYST)
	"prim"	type 1 processing of ties
	"seco"	type 2 processing of ties
	"tert"	type 3 processing of ties
`"trans"`	char	kind of transformation
	"none"	no transformation
	"rati"	same as "none"
	"intr"	interval (see "regres" in MULTISCALE)
	"ordi"	ordinal (see "regres" and "ties"in KYST)
	"powe"	power (see "regres" and "poldeg"in MULTISCALE and KYST)
	"spli"	spline
	"mspl"	M spline (see "regres" in MULTISCALE)

Output: 1. When the `"ndim"` or the `"dimmin"` and `"dimmax"` input options specify only one dimension d, i.e. $nd = 1$, the return arguments are now:

"gof" is a column vector,

"conf" either matrix or tensor is returned:

(a) no `"iase"` option: no standard errors: matrix `xm[n,d]` is returned

(b) with `"iase"` option: with standard errors: tensor `xt[4,n,d]` is returned with estimates, ASEs, LCIs and UCIs in its slides.

"resi" tensor is returned:

(a) for only $m = 1$ input matrix: tensor resi[4,n,n] of residuals is returned, where the first slide corresponds to the input data, the second slide to the preprocessed (transformed) input data, the third slide the fitted distances and the last slide to the residuals (difference between slides 2 and 3)

(b) for $m > 1$: input matrices: tensor resi[m,4,nobs,nobs] is returned

"wgt" for TORGERSON, ALSCAL, COSPA, INDSCAL, SUMSCAL methods

TORGERSON, ALSCAL : missing values are returned

MULTISCALE : matrix or tensor:

(a) for $m = 1$ input matrix: `wgt` is either diagonal or symmetric $d \times d$ matrix

(b) for $m > 1$ input matrices: `wgt` is tensor $m \times d \times d$]

COSPA, INDSCAL, SUMSCAL : matrix or tensor:

(a) for $m = 1$ input matrix: diagonal $d \times d$ matrix

(b) for $m > 1$ input matrices: tensor $m \times d \times d$]

KYST : $n \times d$ matrix of gradients

"grad" for KYST: returns vggradient as $n \times d$ matrix

"eval" for TORGerson, ALSCAL, COSPA, INDSCAL, SUMSCAL only returns vector of d values (or scalar for d=1)

"evec" for TORGerson, ALSCAL, COSPA, INDSCAL, SUMSCAL only

"cov" for MULTISCAL and SMACOF only: returns tensor [n,d,d] of n symmetric $d \times d$ covariance matrices of the $n \times d$ configution, one for each of the n points

2. Whenever the `"dimmin"` and `"dimmax"` input options specify more than one dimensions, $nd > 1$, i.e. $d_1, \ldots, d_v$, then

- the `"gof` return argument becomes a matrix with v columns,
- and all other return arguments become lists of v data objects.

Restrictions: 1. At that time, the data of the second input argument should not contain any missing values, string or complex values. Only real and integer data are permitted.

2.

Relationships: mdu(), dist(), cluster(), anacor(), anaprof()

Examples: Below we illustrate the `mds()` function with the following examples:

1. Individual differences scaling with two normal generated data sets by COSPA, SUMSCAL, and INDSCAL, see `tmds.inp`
2. MDS analysis of randomly generated raw data in matrix form by TORGERSON, ALSCAL, KYST, SMACOF, and MAXLIK
3. Example from the KYST Manual
4. Examples from MULTISCALE Manual
   (a) HANGUL by Yoshio Takane: 3-way Rectangular Data: m=3, n=14
   (b) MDS of dissimilarity judgements among 10 simple vowels: m=3, n=14
5. Test Example: Unidimensional scaling

- COSPA, SUMSCAL, and INDSCAL Test Examples (see `tmds.inp`)
  1. Randomly generated raw data in tensor form `X[2,n,p]`:

```
srand(123);
nind = [ 2 10 10 ]; m = p = nind[1];
nway = size(nind); n = nind[2]; nd = 4;
none = cons(n,1,1.); pone = cons(p,1,1.);

/* [1] Uniform random generation of pxnd weight matrix: W[p,nd] */
wght = rand(p,nd,"g");
print "Uniform: Unscaled wght=",wght;

/* [2] Normal random generated n x nd configuration: X[n,nd] */
coef  = rand(n,nd,"g","norm",0.,1.);
coef += rand(n,nd,"g","norm",1.,1.);
mu = cons(nd,1,1.); id = ide(nd);
for (j = 1; j <= n; j++) {
  coef[j,] += mrand("mnor",mu,id)‘;
}
print "Normal: Unscaled coef=",coef;

/* [3] Normalize data[p+n,nd]: NORCONF() */
/* [3.1] center X[n,nd] */
mu = coef[+,] / n;
```

```
coef = coef - none * mu;
print "Centered Coef=", coef;

/* [3.2] scaling coef[n,nd] */
tv = coef[**,]; rv = 1. ./ sqrt(tv);
coef = coef .* (none * rv);
print "Scaled Coef=", coef;
print "Mean=", coef[+,];
print "Scale=", coef[**,];

/* [3.3] scaling wght[p,nd] */
twgt = wght .* (pone * tv);
print "Scaled WGT=", twgt;

/* [4] Concatenate COEF and WGHT */
wcof = twgt |> coef;

/* [5] Permute dimensions according weights
       sum of squares in descending order */
wscl = twgt[**,];
wrnk = wscl[>:]; print "Ranks=", wrnk;
wcof = wcof[,wrnk];
n1 = p+1; n2 = p+n;
wght = wcof[1:p,]; coef = wcof[n1:n2,];

/* [6] Weighted Configuration: Y = X * W */
real wxdat[p,n,nd];
for (j = 1; j <= p; j++) {
  wv = wght[j,];
  wxdat[j,,] = coef .* (none * wv);
  /* print "WXdat[j,,]=", wxdat[j,,]; */
}
print "Tensor WXDAT=",wxdat;
```

```
Tensor WXDAT=
                  ****************************
                  Tensor wxdat with 3 Dimensions
                  ****************************


                           wxdat_1
                           *******

                      Dense Matrix (10 by 4)
```

```
   |          1          2          3          4
---------------------------------------------------
 1 |  -14.554672 -1.6301023  4.9856908  0.0606599
 2 |  -3.0380098 -4.5080465  1.0150659  0.0439437
 3 |   6.4290345 -7.0272730  0.4288613 -0.6636031
 4 |  -8.4827305  6.9577941 -1.4993646  0.1756061
 5 |   18.784123  1.9255437  3.3502585  0.0904182
 6 |   13.203212  0.4243445 -3.0888729 -0.5465808
 7 |   1.3760075  3.8225599  1.3278726  0.2144417
 8 |  -6.3482351  1.5903091  2.2758083  0.0053984
 9 |  -4.2046344 -5.6754399 -5.2610451  0.3112615
10 |  -3.1640951  4.1203103 -3.5342749  0.3084545

                    wxdat_2
                    *******

             Dense Matrix (10 by 4)

   |          1          2          3          4
---------------------------------------------------
 1 |  -6.3948545 -0.7757735  3.4352285  0.2674217
 2 |  -1.3348037 -2.1454008  0.6993982  0.1937275
 3 |   2.8247109 -3.3443127  0.2954930 -2.9255223
 4 |  -3.7270388  3.3112474 -1.0330885  0.7741671
 5 |   8.2531390  0.9163754  2.3083869  0.3986123
 6 |   5.8010665  0.2019476 -2.1282877 -2.4096247
 7 |   0.6045734  1.8191745  0.9149276  0.9453753
 8 |  -2.7892102  0.7568357  1.5680719  0.0237990
 9 |  -1.8473810 -2.7009689 -3.6249525  1.3722094
10 |  -1.3902016  1.9608753 -2.4351774  1.3598348
```

(a) Analysis with COSPA:

```
optn = [ "data"      "raw" ,
         "ndim"          2 ,
         "norm"           ];
< gof,conf,resi,wgt,eval,evec > = mds("cospa",wxdat,optn);
print "GOF=",gof;
print "X Configuration=",conf;
print "W Configuration=",wgt;
print "Eigenvalues=",eval;
print "Eigenvectors=",evec;

                ****************************
                3-Way Multidimensional Scaling
```

Method= COSPA

Raw Input Data

Number Rows of Matrix	10
Number Columns of Matrix	4
Dimension	2
Number Subjects / Replications	2
Amount Input Information	90
Number of Parameters	24
Degrees of Freedom	66
Maximum Number Iterations	400

Compute Metric SCHOENEMANN COSPA

Nonzero Eigenvalues of Mean Scalar Product Matrix

1 :	0.9573	0.2185	0.1664	0.02963

Estimated Rank : 4

Unrotated common space configuration

	DIM_1	DIM_2
O_01	-0.465592568	0.071152585
O_02	-0.093331284	0.147963476
O_03	0.227242154	0.246678538
O_04	-0.278491720	-0.215828327
O_05	0.585362887	-0.105058979
O_06	0.434267720	-0.008289568
O_07	0.032286775	-0.137844258
O_08	-0.205208492	-0.042378202
O_09	-0.128315170	0.181828134
O_10	-0.108220302	-0.138223400

Undiagonalized Weight Matrices:

Weight matrix: 1

	DIM_1	DIM_2
DIM_1	1.017745116	
DIM_2	-0.027635222	0.907661656

```
      Weight matrix:    2

               DIM_1        DIM_2
DIM_1   0.982254884
DIM_2   0.027635222  1.092338344

   Orthogonal Rotation Matrix

               DIM_1        DIM_2
DIM_1   0.230560993  0.973057875
DIM_2   0.973057875 -0.230560993

Rotated common space configuration

              DIM_1        DIM_2
O_01  -0.038111902 -0.469453525
O_02   0.122458472 -0.124931347
O_03   0.292425671  0.164245319
O_04  -0.274222781 -0.221226968
O_05   0.032733382  0.593814469
O_06   0.092058967  0.424478876
O_07  -0.126686370  0.063198410
O_08  -0.088549516 -0.189908979
O_09   0.147344825 -0.166780562
O_10  -0.159450748 -0.073435693

  Diagonalized Weight Matrices:
  ****************************

      Weight matrix:    1

               DIM_1        DIM_2
DIM_1   0.901113634
DIM_2  -3.8580e-015  1.024293137

      Weight matrix:    2

               DIM_1        DIM_2
DIM_1   1.098886366
DIM_2   2.1927e-015  0.975706863

Schoenemann's Testindizes for COSPA

  Matrix  CommonSpace  Diagonality
```

```
                    1  0.964379639  2.8395e-015
                    2  0.885697342  1.4974e-015

                ******************************
                Result Normalized Configuration
                ******************************

                   Common Space Configuration

                             DIM_1        DIM_2
                O_01  -0.489977791 -0.075071833
                O_02  -0.130393281  0.241215514
                O_03   0.171426039  0.576012484
                O_04  -0.230898897 -0.540156903
                O_05   0.619775732  0.064477364
                O_06   0.443036873  0.181335360
                O_07   0.065961412 -0.249543518
                O_08  -0.198211702 -0.174422535
                O_09  -0.174072122  0.290236005
                O_10  -0.076646263 -0.314081938

                   Subjective Weight Matrices

                             DIM_1        DIM_2
                 W_1   0.940278879  0.232245207
                 W_2   0.895677733  0.283217434

               Goodness-of-fit resp. SCP Matrices
               **********************************

                    N  Av_SSQ_Diff  Correlation
                    1  0.000125092  0.994679763
                    2  0.000660168  0.972705057

            Goodness-of-fit resp. Distance Matrices
            ***************************************

                    N  Av_SSQ_Diff  Correlation
                    1  2.033206077  0.990027529
                    2  1.716943364  0.965981685

GOF=
          |         1
----------------------
    Error |   0.00000
```

```
     Time |   0.00000
 Ind.Inf. |    90.000
   NParms |    24.000
       DF |    66.000
   unused |     .
   unused |     .
   unused |     .
   unused |     .
   unused |     .
```

X Configuration=

	DIM_1	DIM_2
O_01	-0.48998	-0.07507
O_02	-0.13039	0.24122
O_03	0.17143	0.57601
O_04	-0.23090	-0.54016
O_05	0.61978	0.06448
O_06	0.44304	0.18134
O_07	0.06596	-0.24954
O_08	-0.19821	-0.17442
O_09	-0.17407	0.29024
O_10	-0.07665	-0.31408

W Configuration=

	DIM_1	DIM_2
W_1	0.94028	0.23225
W_2	0.89568	0.28322

Eigenvalues=

	DIM_1	DIM_2
1	0.95725	0.21846

Eigenvectors=

	DIM_1	DIM_2
O_01	-0.47588	0.15223
O_02	-0.09539	0.31657
O_03	0.23226	0.52777
O_04	-0.28464	-0.46177
O_05	0.59829	-0.22478
O_06	0.44386	-0.01774
O_07	0.03300	-0.29492

```
O_08 |   -0.20974   -0.09067
O_09 |   -0.13115    0.38902
O_10 |   -0.11061   -0.29573
```

(b) Analysis with SUMSCAL:
We run the same analysis with the SUMSCAL method, but now for two and also one dimension:

```
optn = [ "data"      "raw" ,
         "dimmax"        2 ,
         "dimmin"        1 ,
         "norm"           ];
< gof,conf,resi,wgt,eval,evec > = mds("sumsc",wxdat,optn);
```

```
                ****************************
                3-Way Multidimension Scaling
                ****************************

                      Raw Input Data
 Number Rows of Matrix . . . . . .          10
 Dimension . . . . . .           2
 Amount Input Information. . . . . .        90
 Number of Parameters. . . . . .            24
 Degrees of Freedom. . . . . .              66

              *******************************
              Compute DeLEEUW-PRUZANSKY SUMSCAL
              *******************************

 Eigenvalues of Mean Scalar Product Matrix
 -----------------------------------------

   1 :      0.9573     0.2185     0.1664    0.02963  1.191e-016
   6 :   1.633e-017 -1.537e-017 -4.755e-017 -1.052e-016 -3.128e-016

            Unrotated common space configuration

                       D_2_1        D_2_2
          O_01  -0.465592568  0.071152585
          O_02  -0.093331284  0.147963476
          O_03   0.227242154  0.246678538
          O_04  -0.278491720 -0.215828327
          O_05   0.585362887 -0.105058979
          O_06   0.434267720 -0.008289568
          O_07   0.032286775 -0.137844258
          O_08  -0.205208492 -0.042378202
```

```
O_09  -0.128315170  0.181828134
O_10  -0.108220302 -0.138223400

Undiagonalized Weight Matrices:
*******************************

      Weight matrix:    1

             D_2_1        D_2_2
D_2_1   1.017745116
D_2_2  -0.027635222  0.907661656

      Weight matrix:    2

             D_2_1        D_2_2
D_2_1   0.982254884
D_2_2   0.027635222  1.092338344

 Iterative Simultaneous Diagonalization:
 ***************************************

Iter  ssq-off-dia     max(ang)     min(cos)
   1  0.055270443  1.3408e+154  0.000000000
   2  4.3715e-016  0.232654169  0.973057875
   3  4.3715e-016  2.8166e-017  1.000000000

Residuals SIMDIAG: nit = 2
--------------------------

  1 :   4.627e-032  4.93e-032

                Orthogonal Rotation Matrix

             D_2_1        D_2_2
D_2_1   0.973057875 -0.230560993
D_2_2   0.230560993  0.973057875

              Rotated common space configuration

              D_2_1        D_2_2
O_01  -0.469453525 -0.038111902
O_02  -0.124931347  0.122458472
O_03   0.164245319  0.292425671
O_04  -0.221226968 -0.274222781
O_05   0.593814469  0.032733382
```

```
O_06   0.424478876  0.092058967
O_07   0.063198410 -0.126686370
O_08  -0.189908979 -0.088549516
O_09  -0.166780562  0.147344825
O_10  -0.073435693 -0.159450748

 Diagonalized Weight Matrices:
 *****************************

      Weight matrix:    1

              D_2_1        D_2_2
D_2_1   1.024293137
D_2_2  -2.1511e-016  0.901113634

      Weight matrix:    2

              D_2_1        D_2_2
D_2_1   0.975706863
D_2_2  -2.2204e-016  1.098886366

Schoenemann's Testindizes for SUMSCAL

      1  0.964379639  1.5832e-016
      2  0.885697342  1.5163e-016

  ******************************
  Result Normalized Configuration
  ******************************

     Common Space Configuration

            D_2_1        D_2_2
O_01  -0.489977791 -0.075071833
O_02  -0.130393281  0.241215514
O_03   0.171426039  0.576012484
O_04  -0.230898897 -0.540156903
O_05   0.619775732  0.064477364
O_06   0.443036873  0.181335360
O_07   0.065961412 -0.249543518
O_08  -0.198211702 -0.174422535
O_09  -0.174072122  0.290236005
O_10  -0.076646263 -0.314081938

     Subjective Weight Matrices
```

```
            D_2_1        D_2_2
W_1   0.940278879  0.232245207
W_2   0.895677733  0.283217434


 Goodness-of-fit resp. SCP Matrices
 **********************************

      N  Av_SSQ_Diff  Correlation
      1  0.000125092  0.994679763
      2  0.000660168  0.972705057

Goodness-of-fit resp. Distance Matrices
***************************************

      N  Av_SSQ_Diff  Correlation
      1  2.033206077  0.990027529
      2  1.716943364  0.965981685
```

It follows the analysis in one dimension:

```
**************************************************
Compute DeLeeuw-Pruzansky SUMSCAL for 1 Dimension(s)
**************************************************

   Unrotated common space configuration

              D_1_1
O_01  -0.465592568
O_02  -0.093331284
O_03   0.227242154
O_04  -0.278491720
O_05   0.585362887
O_06   0.434267720
O_07   0.032286775
O_08  -0.205208492
O_09  -0.128315170
O_10  -0.108220302

  Undiagonalized Weight Matrices:
  *******************************

      Weight matrix:    1

              D_1_1
```

```
          D_1_1   1.017745116

          Weight matrix:    2

                       D_1_1
          D_1_1   0.982254884

  Schoenemann's Testindizes for SUMSCAL

          1  0.908403857  0.000000000
          2  0.816123639  0.000000000

     ******************************
     Result Normalized Configuration
     ******************************

        Common Space Configuration

                       D_1_1
          O_01  -0.475875172
          O_02  -0.095392504
          O_03   0.232260793
          O_04  -0.284642206
          O_05   0.598290617
          O_06   0.443858516
          O_07   0.032999828
          O_08  -0.209740518
          O_09  -0.131149009
          O_10  -0.110610345

        Subjective Weight Matrices

                      D_1_1
          W_1   0.974237878
          W_2   0.940264806

    Goodness-of-fit resp. SCP Matrices
    **********************************

          N  Av_SSQ_Diff  Correlation
          1  0.000574718  0.975346430
          2  0.001385490  0.941856456

  Goodness-of-fit resp. Distance Matrices
  ***************************************
```

```
                        N  Av_SSQ_Diff  Correlation
                        1  0.000000000  1.000000000
                        2  0.000000000  1.000000000
```

(c) Analysis with INDSCAL:

```
optn = [ "data"      "raw" ,
         "ndim"          2 ,
         "print"         4 ,
         "norm"           ];
< gof,conf,resi,wgt,eval,evec > = mds("indsc",wxdat,optn);

                  ****************************
                  3-Way Multidimensional Scaling
                         Method= INDSCAL
                  ****************************

                          Raw Input Data
 Number Rows of Matrix . . . . . .          10
 Number Columns of Matrix  . . . . .         4
 Dimension . . . . . .                       2
 Number Subjects / Replications. . . . .     2
 Amount Input Information. . . . . .        90
 Number of Parameters. . . . . .            24
 Degrees of Freedom. . . . . .              66
 Maximum Number Iterations . . . . . .    5000

              ************************************
              Compute Metric CARROLL-CHANG CANDECOMP
              ************************************

 Nonzero Eigenvalues of Mean Scalar Product Matrix
 -------------------------------------------------

   1 :      0.9573     0.2185     0.1664    0.02963
                       Estimated Rank : 4
```

The following output of the iteration history could be restricted to larger steps using the `plin` option (default is 10):

```
                  Iterative Canonical Decomposition:
                  **********************************

           Iter  Way    Max. Dev.     Maximum     Residual
              1    1  0.092338344
              1    2  0.002573734  0.092338344  0.275733651
```

```
 11    1  0.000126273
 11    2  0.001420932  0.001420932  0.275619458
 21    1  9.1720e-005
 21    2  0.001341886  0.001341886  0.275430703
 31    1  5.9351e-005
 31    2  0.001262113  0.001262113  0.275255123
 41    1  3.3093e-005
 41    2  0.001178954  0.001178954  0.275097042
  ..........................................
631    1  3.3370e-006
631    2  1.2614e-005  1.2614e-005  0.274047651
641    1  3.1349e-006
641    2  1.1837e-005  1.1837e-005  0.274047522
651    1  2.9452e-006
651    2  1.1108e-005  1.1108e-005  0.274047402
661    1  2.7671e-006
661    2  1.0426e-005  1.0426e-005  0.274047292
664    0  9.9616e-006  9.9616e-006  0.274047235

Residuals CADECO: nit = 664
---------------------------

  1 :    0.006999    0.03625

            Result non-normalized configuration X:
            **************************************

                 Common Space Configuration

                          DIM_1        DIM_2
               O_01  -0.469705827 -0.036553819
               O_02  -0.124329431  0.121437948
               O_03   0.164347235  0.294394706
               O_04  -0.222344182 -0.271971552
               O_05   0.594691055  0.028653940
               O_06   0.423981427  0.094048475
               O_07   0.063142821 -0.127689246
               O_08  -0.190385558 -0.087104099
               O_09  -0.165710781  0.144643058
               O_10  -0.073686760 -0.159859411

                 Common Space Configuration

                          DIM_1        DIM_2
               O_01  -0.469705827 -0.036553819
```

```
O_02  -0.124329431  0.121437948
O_03   0.164347235  0.294394706
O_04  -0.222344182 -0.271971552
O_05   0.594691055  0.028653940
O_06   0.423981427  0.094048475
O_07   0.063142821 -0.127689246
O_08  -0.190385558 -0.087104099
O_09  -0.165710781  0.144643058
O_10  -0.073686760 -0.159859411
```

Subjective Weight Matrices

```
            DIM_1        DIM_2
 W_1   1.023979211  0.899153581
 W_2   0.975339995  1.102091165
```

Result Normalized Configuration

****

Common Space Configuration

```
            DIM_1        DIM_2
O_01  -0.489950010 -0.072139585
O_02  -0.129687993  0.239659861
O_03   0.171430553  0.580992970
O_04  -0.231927151 -0.536740493
O_05   0.620322065  0.056549038
O_06   0.442254902  0.185606268
O_07   0.065864258 -0.251996903
O_08  -0.198591120 -0.171901425
O_09  -0.172852867  0.285455541
O_10  -0.076862638 -0.315485272
```

Common Space Configuration

```
            DIM_1        DIM_2
O_01  -0.489950010 -0.072139585
O_02  -0.129687993  0.239659861
O_03   0.171430553  0.580992970
O_04  -0.231927151 -0.536740493
O_05   0.620322065  0.056549038
O_06   0.442254902  0.185606268
O_07   0.065864258 -0.251996903
O_08  -0.198591120 -0.171901425
O_09  -0.172852867  0.285455541
```

```
O_10  -0.076862638 -0.315485272

   Subjective Weight Matrices

             DIM_1        DIM_2
W_1   0.941108062  0.230861820
W_2   0.896405242  0.282966978

Goodness-of-fit resp. SCP Matrices
**********************************

     N  Av_SSQ_Diff  Correlation
     1  0.000127246  0.994584218
     2  0.000659062  0.972744353

Goodness-of-fit resp. Distance Matrices
***************************************

     N  Av_SSQ_Diff  Correlation
     1  2.075138105  0.989819063
     2  1.718323923  0.965938432
```

2. Scalar product and distance data in tensor form `D[2,n,n]`:
   The first part of randomly generating the `coef` and `wgt` data is the same as for the raw data tensor `wxdat` above.
   The tensor data `eudis` of distances are then obtained using the `dist` function:

```
/* [6] Weighted Euclidean Distances: WEUCDIS() */
real eudis[p,n,n];
for (j = 1; j <= p; j++) {
  wv = wght[j,];
  xcof = coef .* (none * wv);
  optn = [ 0, 0 ];
  eudis[j,,] = dist(xcof,"L2",optn);
  /* print "Eudis[j,,]=", eudis[j,,]; */
}
print "Tensor EUDIS=",eudis;
```

```
Tensor EUDIS=
                  ****************************
                  Tensor eudis with 3 Dimensions
                  ****************************

                            eudis_1
```

****

Dense Symmetric Matrix (10 by 10)

S	1	2	3	4	5
1	0.0000000				
2	12.517277	0.0000000			
3	22.152531	9.8394964	0.0000000		
4	12.356763	12.940254	20.551525	0.0000000	
5	33.567743	22.870327	15.553271	28.148374	0.0000000
6	28.987725	17.472745	10.667907	22.715945	8.6757329
7	17.231406	9.4344830	12.034581	10.724700	17.628005
8	9.2229700	7.0525519	15.536413	6.9026711	25.157692
9	15.117752	6.4949865	12.174883	13.858909	25.699440
10	15.344827	9.7586323	15.262637	6.3637807	23.108129

S	6	7	8	9	10
6	0.0000000				
7	13.096467	0.0000000			
8	20.315094	8.0987128	0.0000000		
9	18.592868	12.836619	10.690362	0.0000000	
10	16.807093	6.6596174	7.0984902	10.001059	0.0000000

eudis_2

****

Dense Symmetric Matrix (10 by 10)

S	1	2	3	4	5
1	0.0000000				
2	5.9135599	0.0000000			
3	10.566485	5.3508700	0.0000000		
4	6.6365373	6.2318712	10.132854	0.0000000	
5	14.788985	10.194800	7.9197796	12.671486	0.0000000
6	13.704587	8.4382427	5.2512397	10.573009	5.8389505
7	7.9080518	4.4822392	6.8526023	4.9812874	7.8457810
8	4.3468595	3.3647800	7.6585561	3.8384410	11.074632
9	8.6863458	4.5453515	7.4888400	6.8377826	12.298703
10	8.2577739	5.2962197	8.4694364	3.0972848	10.840219

S	6	7	8	9	10

```
     6 |   0.0000000
     7 |   7.0806884  0.0000000
     8 |   9.6791208  3.7312506  0.0000000
     9 |   9.1360795  6.8728797  6.4520669  0.0000000
    10 |   8.3133128  3.9235426  4.6062928  4.8329621  0.0000000
```

All the results obtained with COSPA, SUMSCAL, and INDSCAL are the same as those obtained with the raw data input. Here we only show the `mds` call for COSPA:

```
optn = [ "data"      "dis" ,
         "nobs"         10 ,
         "ndim"          2 ,
         "norm"           ];
< gof,conf,resi,wgt,eval,evec > = mds("cospa",eudis,optn);
print "GOF=",gof;
print "X Configuration=",conf;
print "W Configuration=",wgt;
print "Eigenvalues=",eval;
print "Eigenvectors=",evec;
```

The tensor data `scalp` of scalar products are then obtained using the simple code:

```
/* [6] Weighted Configuration: Y = X * W */
real scalp[p,n,n];
for (j = 1; j <= p; j++) {
  wv = wght[j,];
  xdat = coef .* (none * wv);
  /* print "Xdat=", xdat; */
  scalp[j,,] = xdat * xdat';
}
print "Tensor SCALP=",scalp;
```

```
Tensor SCALP=
                  ****************************
                  Tensor scalp with 3 Dimensions
                  ****************************


                           scalp_1
                           *******

                Dense Symmetric Matrix (10 by 10)

      S |          1          2          3          4          5
      ---------------------------------------------------------------
```

1	239.35651				
2	56.629284	30.584276			
3	-80.019400	12.553964	91.339342		
4	104.65673	-7.1096775	-104.18964	122.64655	
5	-259.82674	-62.342084	108.60925	-150.95050	367.78340
6	-208.29347	-45.183882	80.939938	-104.51143	238.42991
7	-19.625132	-20.055300	-17.588603	12.970976	37.675707
8	101.15092	14.427146	-51.016135	61.504092	-108.55878
9	44.237560	33.032238	10.388312	4.1211227	-107.50639
10	21.733749	-12.535965	-51.017027	60.861770	-63.313758

S	6	7	8	9	10
6	184.34477				
7	15.570962	18.314592			
8	-90.174890	0.3669736	48.008505		
9	-41.842453	-34.399568	5.6948546	77.665048	
10	-29.279458	6.7693928	18.597319	8.6092796	39.574698

scalp_2
****

Dense Symmetric Matrix (10 by 10)

S	1	2	3	4	5
1	53.368298				
2	12.654620	6.9111339			
3	-15.236448	3.0443698	27.809416		
4	17.923227	-2.7016504	-24.171767	26.521784	
5	-45.452089	-11.290610	19.764048	-29.801599	74.441589
6	-45.209182	-10.131875	22.131477	-20.618844	42.188651
7	-1.8816273	-3.8868014	-6.8715051	3.5571455	9.1455204
8	22.642510	3.2006496	-10.016078	11.300032	-18.696990
9	1.8234916	5.9911013	-1.2710098	2.7489006	-25.542608
10	-0.6327998	-3.7909387	-15.182503	15.242772	-14.755914

S	6	7	8	9	10
6	44.029056				
7	-0.3506802	5.4057317			
8	-19.422208	1.1477050	10.811910		
9	-6.8538026	-8.0497273	-2.5429844	25.731289	
10	-5.7625912	1.7842387	1.5754542	7.9653492	13.556932

Running the following call of `mds` will generate the same results as reported above for other data input:

```
optn = [ "data"      "scp" ,
         "nobs"         10 ,
         "ndim"          2 ,
         "norm"           ];
< gof,conf,resi,wgt,eval,evec > = mds("cospa",scalp,optn);
```

- TORGERSON, ALSCAL, KYST, SMACOF, and MAXLIK Test Examples

  1. Randomly generated raw data in matrix form `X[n,p]`:

```
srand(123);

/* Normal random generated nxnd configuration: X[n,nd] */
n = 10; nd = 4;
coef  = rand(n,nd,"g","norm",0.,1.);
coef += rand(n,nd,"g","norm",1.,1.);
mu = cons(nd,1,1.); id = ide(nd);
for (j = 1; j <= n; j++) coef[j,] += mrand("mnor",mu,id)‘;
print "Normal: Unscaled coef=",coef;
data = dist(coef,"l2");
print "EuclDistance=", data;
```

```
Normal: Unscaled coef=
    |         1         2         3         4
-------------------------------------------
  1 |   0.13945    5.6162    4.9593   0.89268
  2 |   0.46138    3.9600    2.1125  -0.30472
  3 |   0.24811    3.6825  -0.03017    3.1384
  4 |    3.0081   0.83865    3.5334  -0.26173
  5 |   0.38527    1.5628    2.9947    3.1497
  6 |    1.3847    2.1498   0.19754    4.5737
  7 |    3.0734    3.7302    2.1824    3.4404
  8 |    2.0061    3.5394    3.2046    2.3395
  9 |   0.69489    1.5902    1.5327   -1.9894
 10 |    2.1309    1.0724    3.1897    2.5358

EuclDistance=
  S |         1         2         3         4         5
-----------------------------------------------------------
  1 |   0.00000
  2 |    3.5192   0.00000
  3 |    5.8043    4.0705   0.00000
  4 |    5.8669    4.2719    6.3218   0.00000
```

```
  5 |    5.0443    4.2970    3.6962    4.3968   0.00000
  6 |    7.0563    5.6210    2.3985    6.2341    3.3459
  7 |    5.1349    4.5723    3.6016    4.8883    3.5592
  8 |    3.6014    3.2783    3.7700    3.8952    2.6897
  9 |    6.0469    2.9740    5.7718    3.5921    5.3521
 10 |    5.5176    4.5115    4.5921    2.9612    1.9242

  S |         6         7         8         9        10
----------------------------------------------------------
  6 |   0.00000
  7 |    3.2517   0.00000
  8 |    4.0437    1.8527   0.00000
  9 |    6.7562    6.3357    5.2012   0.00000
 10 |    3.8501    3.1281    2.4780    5.0550   0.00000
```

2. Symmetric metric Torgerson analysis:

```
print "Torgerson MDS with ";
optn = [ "data"        "raw" ,
         "dimmax"          2 ,
         "noplot"            ,
         "print"           4 ,
         "pres"              ,
         "ptab"              ,
         "phis"            1 ];
< gof,conf,resi,wgt > = mds("torg",coef,optn);
print "GOF=",gof;
print "X Configuration=",conf;
print "Resi=", resi;
print "WGT=", wgt;
```

```
                ******************************
                Metric Multidimensional Scaling
                       Method= TORGERSON
                ******************************

                          Raw Input Data
   Number Rows of Matrix . . . . . .          10
   Number Columns of Matrix  . . . . .         4
   Maximum Dimension . . . . . .               2
   Minimum Dimension . . . . . .               1
   Amount Input Information. . . . . .        45
   Number of Parameters. . . . . .            20
   Degrees of Freedom. . . . . .              25
   Maximum Number Iterations . . . . . .       0
```

Distance Matrix: Subject=1

	O_01	O_02	O_03	O_04
O_01	0			
O_02	3.519200911	0		
O_03	5.804268075	4.070462971	0	
O_04	5.866913497	4.271892779	6.321752515	0
O_05	5.044342860	4.296962987	3.696197626	4.396815536
O_06	7.056315012	5.621013930	2.398542813	6.234111864
O_07	5.134872756	4.572298040	3.601588486	4.888345013
O_08	3.601364129	3.278305144	3.769988407	3.895202096
O_09	6.046922293	2.973986513	5.771839152	3.592116665
O_10	5.517579023	4.511542190	4.592145078	2.961171029

Distance Matrix: Subject=1

	O_05	O_06	O_07	O_08
O_05	0			
O_06	3.345921485	0		
O_07	3.559243240	3.251748876	0	
O_08	2.689730422	4.043676200	1.852666721	0
O_09	5.352087666	6.756239769	6.335731266	5.201245255
O_10	1.924202616	3.850144991	3.128098230	2.477967633

Distance Matrix: Subject=1

	O_09	O_10
O_09	0	
O_10	5.055034646	0

Additive Constant: 0

Estimated X Configuration Matrix

	D_2_1	D_2_2
O_01	17.83722960	-1.496037725
O_02	10.49510347	-1.664731028
O_03	12.78374483	2.352556296
O_04	14.24789852	-2.013708442
O_05	9.167882940	1.124865363
O_06	15.10779794	3.668654160
O_07	10.59712527	1.912106275
O_08	7.263401404	0.387565530
O_09	17.63432931	-2.908335726
O_10	9.186465018	0.570848889

```
Sorted SSQ(X)
-------------

  1 :       125.7      37.76

                  Final X Configuration Matrix

                              D_2_1        D_2_2
                O_01    5.605589416  0.804327969
                O_02   -0.947379985 -2.511371899
                O_03   -0.612983175  2.099989667
                O_04    2.589565335 -1.209345134
                O_05   -3.347025918 -0.565848382
                O_06    0.917218924  4.289011825
                O_07   -2.396449383  0.760381671
                O_08   -4.747983890 -2.051774937
                O_09    6.030999285 -0.557575337
                O_10   -3.091550607 -1.057795444
```

3. Symmetric metric ALSCAL analysis:

```
print "ALSCAL MDS with ";
optn = [ "data"        "raw" ,
         "dimmax"          2 ,
         "noplot"            ,
         "print"           4 ,
         "pres"              ,
         "ptab"              ,
         "phis"            1 ];
< gof,conf,resi,wgt,eval,evec > = mds("alsc",coef,optn);
print "GOF=",gof;
print "X Configuration=",conf;
print "Resi=", resi;
print "WGT=", wgt;
print "Eval=", eval;
print "Evec=", evec;
```

```
                ******************************
                Metric Multidimensional Scaling
                         Method= ALSCAL
                ******************************

                          Raw Input Data
 Number Rows of Matrix . . . . . .          10
 Number Columns of Matrix  . . . . .         4
 Maximum Dimension . . . . . .               2
```

```
Minimum Dimension . . . . . .          1
Amount Input Information. . . . . .          45
Number of Parameters. . . . . .          20
Degrees of Freedom. . . . . .          25
Maximum Number Iterations . . . . . .       1000

  ******************************************************
  Compute ALSCAL MDS by Takane, Young, and de Leeuw, 1977
  ******************************************************

  ********** Results for ALSCAL : Dimension 2 **********

          Iteration       Stress   DiffStress

                1  5568.406882  43463.41811
                2  2512.411087  3055.995796
                3  2103.851480  408.5596067
                4  1812.801961  291.0495186
                5  1584.101683  228.7002788
                6  1472.530669  111.5710135
                7  1423.874155  48.65651412
                8  1402.815611  21.05854349
                9  1394.081760  8.733851375
               10  1390.755899  3.325860757
               11  1389.575707  1.180192408
               12  1389.176516  0.399191098
               13  1389.046087  0.130428548
               14  1389.004587  0.041500156
               15  1388.991638  0.012949148
               16  1388.987651  0.003986547
               17  1388.986435  0.001216657
               18  1388.986065  0.000369315
               19  1388.985954  0.000111746
               20  1388.985920  3.3749e-005

         Estimated X Configuration Matrix

                    D_2_1        D_2_2
        O_01   16.54888492 -0.354495174
        O_02   14.60274628 -1.456706724
        O_03   12.54428247  2.590008491
        O_04   11.28483434 -3.422170238
        O_05   10.95930509  0.382121745
        O_06   10.11575516  2.482135468
        O_07   12.05914448  1.937432869
```

```
                O_08   13.15213344  0.379981921
                O_09   12.72645262 -4.181664994
                O_10   10.49555915 -0.698728250
 Configuration X is centered and rotated: Varimax (Kaiser, 1958).

 Sorted SSQ(X)
 -------------

   1 :       47.48      35.84

                  Final X Configuration Matrix

                              D_2_1        D_2_2
                O_01  -0.109001539  4.100290672
                O_02   1.100235117  2.218782597
                O_03  -2.825133640 -0.062589228
                O_04   3.248027550 -0.984118289
                O_05  -0.532130453 -1.521716898
                O_06  -2.581726800 -2.481294378
                O_07  -2.146468853 -0.510504311
                O_08  -0.652525839  0.667804905
                O_09   3.925780249  0.497686967
                O_10   0.572944206 -1.924342037

GOF=
           COL |         1
---------------------------
   1 Error     |   0.00000
   2 Time      |   0.00000
   3 Nobs      |   10.0000
   4 Ndim      |    2.0000
   5 NParm     |    20.000
   6 DF        |    25.000
   7 Stress    |    61.856
   8 N_Iter    |    20.000
   9 unused    |     .
  10 unused    |     .
```

4. Symmetric metric KYST analysis:

```
print "KYST MDS  dim=2";
optn = [ "data"        "raw" ,
         "dimmax"          2 ,
         "noplot"            ,
         "print"           4 ,
         "pres"              ,
```

```
        "ptab"              ,
        "phis"            1 ];
< gof,conf,resi,grad > = mds("kyst",coef,optn);
print "GOF=",gof;
print "X Configuration=",conf;
print "Resi=", resi;
print "Grad=", grad;
print "Data=", data = dist(conf,"l2");
```

```
              ******************************
              Metric Multidimensional Scaling
                       Method= KYST
              ******************************
```

Raw Input Data

Number Rows of Matrix	10
Number Columns of Matrix	4
Dimension	2
Amount Input Information	55
Number of Parameters	20
Degrees of Freedom	35
Maximum Number Iterations	5000
Stress Formula	1
Diagonal	Included
Isotone Regression	Ascending
Tie Treatment	Primary
TORSCA Preiterations	1

Distance Matrix: Subject=1

	O_01	O_02	O_03	O_04
O_01	0			
O_02	3.519200911	0		
O_03	5.804268075	4.070462971	0	
O_04	5.866913497	4.271892779	6.321752515	0
O_05	5.044342860	4.296962987	3.696197626	4.396815536
O_06	7.056315012	5.621013930	2.398542813	6.234111864
O_07	5.134872756	4.572298040	3.601588486	4.888345013
O_08	3.601364129	3.278305144	3.769988407	3.895202096
O_09	6.046922293	2.973986513	5.771839152	3.592116665
O_10	5.517579023	4.511542190	4.592145078	2.961171029

Distance Matrix: Subject=1

O_05	O_06	O_07	O_08

```
O_05            0
O_06   3.345921485            0
O_07   3.559243240  3.251748876            0
O_08   2.689730422  4.043676200  1.852666721            0
O_09   5.352087666  6.756239769  6.335731266  5.201245255
O_10   1.924202616  3.850144991  3.128098230  2.477967633
```

Distance Matrix: Subject=1

```
                 O_09         O_10
O_09                0
O_10   5.055034646            0
```

Additive Constant: 0

History of Iteration for 2 Dimensions

****

Iter	STRESS	SRAT	SRATAV	CAGRCL	COSAV	ACSAV	SFGR	STEP
0	0.130	0.800	0.800	-0.000	-0.000	0.000	0.0058	0.0187
1	0.122	0.939	0.844	0.998	0.659	0.659	0.0048	0.0507
2	0.105	0.866	0.851	0.951	0.852	0.852	0.0026	0.1698
3	0.117	1.111	0.930	-0.852	-0.273	0.852	0.0046	0.0695
4	0.098	0.841	0.899	0.828	0.453	0.836	0.0014	0.0870
5	0.102	1.034	0.942	-0.945	-0.470	0.908	0.0024	0.0290
6	0.097	0.951	0.945	0.969	0.480	0.948	0.0010	0.0342
7	0.096	0.992	0.961	-0.888	-0.423	0.909	0.0007	0.0112
8	0.096	0.996	0.972	0.578	0.238	0.690	0.0001	0.0092
9	0.096	1.001	0.982	-0.903	-0.515	0.831	0.0003	0.0029
10	0.096	0.999	0.987	0.991	0.479	0.936	0.0002	0.0031
11	0.096	1.000	0.992	0.644	0.588	0.743	0.0000	0.0050
12	0.096	1.000	0.994	-0.790	-0.321	0.774	0.0002	0.0018
13	0.096	1.000	0.996	0.973	0.533	0.905	0.0001	0.0022
14	0.096	1.000	0.997	-0.779	-0.333	0.822	0.0000	0.0008
15	0.096	1.000	0.998	-0.807	-0.646	0.812	0.0000	0.0002
16	0.096	1.000	0.999	0.711	0.250	0.745	0.0000	0.0002
17	0.096	1.000	0.999	0.139	0.177	0.345	0.0000	0.0001

Minimum was achieved after 17 iterations.

Estimated X Configuration Matrix

```
                D_1          D_2
O_01  -0.636093956  1.272290263
O_02  -0.742623980  0.376578210
O_03   1.017365014  0.215961450
```

```
                O_04  -0.688267440 -0.922053005
                O_05   0.411052274 -0.331806106
                O_06   1.271510027 -0.255346864
                O_07   0.503330587  0.327031199
                O_08   0.031112472  0.396883611
                O_09  -1.332984281 -0.465443002
                O_10   0.165599283 -0.614095756
Configuration X is centered and rotated: Varimax (Kaiser, 1958).

Sorted SSQ(X)
-------------

  1 :        5.94       4.06

       Final Configuration (Stress= 0.0956301 Formula 1)

                O_01         O_02         O_03         O_04
    D_1   0.111562983  0.546769964 -1.023979646  0.984232681
    D_2   1.418059144  0.627968256 -0.182034624 -0.595969666

       Final Configuration (Stress= 0.0956301 Formula 1)

                O_05         O_06         O_07         O_08
    D_1  -0.256301689 -1.082461276 -0.589317429 -0.177925342
    D_2  -0.461918508 -0.714295005  0.114000235  0.356128010

       Final Configuration (Stress= 0.0956301 Formula 1)

                             O_09         O_10
                D_1   1.410201896  0.077217858
                D_2   0.069389430 -0.631327271

              Group Count      Stress Regression

                  1    55  0.09563009[0]   2.444

                   Gradients at Convergence

                              D_1          D_2
                O_01   1.2661e-006 -2.0329e-006
                O_02  -2.0329e-006  5.0383e-006
                O_03   5.0383e-006  2.4553e-006
                O_04   2.4553e-006 -2.4738e-006
                O_05  -2.4738e-006 -3.1403e-006
                O_06  -3.1403e-006  1.6247e-006
```

```
                O_07   1.6247e-006  2.4697e-006
                O_08   2.4697e-006 -3.3591e-006
                O_09  -3.3591e-006 -1.8479e-006
                O_10  -1.8479e-006 -2.2022e-006

GOF=
           COL |         1
---------------------------
   1 Error     |   0.00000
   2 Time      |   0.00000
   3 Nobs      |   10.0000
   4 Ndim      |    2.0000
   5 NParm     |    20.000
   6 DF        |    35.000
   7 Stress    |    0.0956
   8 N_Iter    |    17.000
   9 unused    |     .
  10 unused    |     .
```

5. Symmetric metric SMACOF analysis:

```
print "SMACOF: iterative relaxation: 37 Iter";
optn = [ "data"        "raw" ,
         "dimmax"          2 ,
         "ftol"        1.e-6 ,
         "relax"             ,
         "noplot"            ,
         "print"           4 ,
         "pres"              ,
         "ptab"              ,
         "phis"            1 ];
< gof,conf,resi,wgt,covm > = mds("smac",coef,optn);
print "GOF=",gof;
print "X Configuration=",conf;
print "Residual=", resi;
print "Wgt=",wgt;
print "COVM=",covm;
```

6. Symmetric metric MAXLIK analysis: Rank deficient design by Ramsay

```
print "MAXLIKelihood MDS: Rank deficient design by Ramsay";
optn = [ "data"        "raw" ,
         "dimmax"          2 ,
         "varobs"      "con" ,
```

```
          "design"     "rams" ,
          "noplot"            ,
          "print"           4 ,
          "pcoef"             ,
          "pdat"              ,
          "pcov"              ,
          "pcor"              ,
          "pres"              ,
          "ptab"              ,
          "phis"            2 ];
< gof,conf,resi,wgt,covm > = mds("mlik",coef,optn);
 print "GOF=",gof;
 print "X Configuration=",conf;
 print "Resi=", resi;
 print "WGT=", wgt;
 print "COVM=", covm;
```

7. Symmetric metric MAXLIK analysis: Full rank design

```
 print "MAXLIKelihood MDS: Full Rank Design";
 optn = [ "data"        "raw" ,
          "dimmax"          2 ,
          "design"     "full" ,
          "varobs"     "con" ,
          "noplot"            ,
          "print"           2 ,
          "pcoef"             ,
          "pdat"              ,
          "pcov"              ,
          "pcor"              ,
          "pres"              ,
          "ptab"              ,
          "phis"            2 ];
< gof,conf,resi,wgt,covm > = mds("mlik",coef,optn);
 print "GOF=",gof;
 print "X Configuration=",conf;
 print "Resi=", resi;
 print "WGT=", wgt;
 print "COVM=", covm;
```

- Test Example from the KYST Manual:

1. Unsymmetric Matrix of Similarity Measures: Regression: Descending

First we use descending isotone (monotone) regression for the relationship among data and model values:

```
simi = [ 9.30  0.57  6.10  2.00  1.50 ,
         1.00  8.50  5.50  3.30  4.00 ,
         2.10  5.30  7.30  3.90  1.10 ,
         5.00  6.20  1.60  8.90  2.80 ,
         1.30  4.10  1.00  2.50  8.80 ];

print "KYST MDS without initial values: Descending";
optn = [ "data"       "simi" ,
         "unsym"             ,
         "regres"     "desc" ,
         "pre-it"          3 ,
         "coord"     "rotat" ,
         "dimmax"          3 ,
         "dimmin"          1 ,
         "phis"              ,
         "norm"             ];
< gof,conf,resi > = mds("kyst",simi,optn);

              *********************************
              NonMetric Multidimensional Scaling
              *********************************

             Similarity Input Data (Unsymmetric)
 Number Rows of Matrix . . . . . .          5
 Maximum Dimension . . . . . .              3
 Minimum Dimension . . . . . .              1
 Amount Input Information. . . . . .       25
 Number of Parameters. . . . . .           15
 Degrees of Freedom. . . . . .             10
 Stress Formula. . . . . .                  1
 Diagonal  . . . . . .                 Included
 Isotone Regression  . . . . . .    Descending
 Tie Treatment . . . . . .             Primary
 Maximum Number Iterations . . . . .      5000
 TORSCA Preiterations. . . . . .           3

Similarity Matrix: 1
--------------------

  1|        9.3       0.57        6.1          2        1.5
  2|          1        8.5        5.5        3.3          4
  3|        2.1        5.3        7.3        3.9        1.1
  4|          5        6.2        1.6        8.9        2.8
```

```
   5|         1.3         4.1           1         2.5         8.8

          ****************************************
          Compute Kruskal-Young-Shepard-Torgerson KYST
          ****************************************
         There are 25 data values, split into 1 lists.

Torgerson-Young Quasi-Nonmetric Procedure for Initial Configuration

                     Iteration      Stress

                             0  0.022638560
                             1  0.002207875
                             2  0.000459813
                             3  0.000153818
 [note] file tmds1.inp, line 63: Maximum number of pre-iterations 3
          reached.
The best initial configuration by Torgerson-Young quasi-nonmetric
method has Stress=0.000153818 Formula 1

              History of Iteration for 3 Dimensions
              ************************************

   Iter STRESS   SRAT SRATAV CAGRCL  COSAV  ACSAV    SFGR    STEP

      0  0.010  0.800  0.800  0.000  0.000  0.000  0.0010  0.0002
        Satisfactory Stress was reached after 0 iterations.

         Final Configuration (Stress= 0.00989601 Formula 1)

                      D_3_1        D_3_2        D_3_3
            O_1   1.075123487 -0.513110454 -0.160950519
            O_2  -0.696452481  0.652971078 -0.001235367
            O_3   0.568985138  0.719528505 -0.156694064
            O_4   0.008885202 -0.233080220  0.586641998
            O_5  -0.956541346 -0.626308909 -0.267762048

               Group Count      Stress Regression

                   1    25  0.00989601 Descending

              History of Iteration for 2 Dimensions
              ************************************

   Iter STRESS   SRAT SRATAV CAGRCL  COSAV  ACSAV    SFGR    STEP
```

```
  0  0.059  0.800  0.800  0.000  0.000  0.000  0.0039  0.0057
  1  0.057  0.968  0.852  0.999  0.659  0.659  0.0037  0.0155
  2  0.052  0.916  0.873  0.988  0.876  0.876  0.0031  0.0530
  3  0.041  0.785  0.843  0.357  0.533  0.533  0.0014  0.1104
     0.060                -0.658                      0.0110
  4  0.039  0.966  0.882  0.757  0.681  0.681  0.0010  0.0291
  5  0.037  0.948  0.903  0.216  0.374  0.374  0.0005  0.0476
     0.051                -0.940                      0.0048
  6  0.037  0.996  0.933 -0.490 -0.196  0.451  0.0002  0.0027
  7  0.037  0.999  0.955 -0.283 -0.253  0.340  0.0002  0.0016
  8  0.037  0.999  0.969  0.007 -0.081  0.120  0.0001  0.0012
  9  0.037  1.000  0.979  0.219  0.117  0.185  0.0001  0.0011
 10  0.037  1.000  0.986 -0.272 -0.140  0.242  0.0001  0.0007
 11  0.037  1.000  0.991 -0.381 -0.299  0.334  0.0000  0.0003
 12  0.037  1.000  0.994  0.111 -0.028  0.187  0.0000  0.0002
 13  0.037  1.000  0.996  0.415  0.264  0.337  0.0000  0.0003
 14  0.037  1.000  0.997  0.008  0.095  0.120  0.0000  0.0002
 15  0.037  1.000  0.998 -0.704 -0.432  0.505  0.0000  0.0001
 16  0.037  1.000  0.999  0.850  0.414  0.733  0.0000  0.0001
 17  0.037  1.000  0.999 -0.496 -0.187  0.577  0.0000  0.0000
        Minimum was achieved after 17 iterations.

    Final Configuration (Stress= 0.0371116 Formula 1)

                        D_2_1        D_2_2
              O_1   1.147921385 -0.496658734
              O_2  -0.646499738  0.672365556
              O_3   0.649763151  0.672655794
              O_4  -0.001631430 -0.354985561
              O_5  -1.149553368 -0.493377055

           Group Count      Stress Regression

               1    25  0.03711159 Descending

          History of Iteration for 1 Dimensions
          *************************************

Iter STRESS   SRAT SRATAV CAGRCL  COSAV  ACSAV    SFGR    STEP

  0  0.212  0.800  0.800  0.000  0.000  0.000  0.0109  0.0575
  1  0.204  0.963  0.851  0.850  0.561  0.561  0.0006  0.1358
  2  0.231  1.134  0.937 -0.946 -0.434  0.815  0.0187  0.0501
  3  0.215  0.928  0.934  0.989  0.505  0.930  0.0110  0.0639
```

```
 4  0.205  0.956  0.941  0.501  0.502  0.647  0.0029  0.1008
    0.213               -0.979                        0.0101
 5  0.205  0.997  0.959  0.997  0.829  0.878  0.0022  0.0267
 6  0.204  0.997  0.972 -0.903 -0.314  0.895  0.0008  0.0094
    0.204               -0.977                        0.0009
 7  0.204  1.000  0.981  1.000  0.553  0.964  0.0006  0.0012
 8  0.204  1.000  0.987  0.999  0.847  0.987  0.0004  0.0028
 9  0.204  1.000  0.991 -0.649 -0.140  0.764  0.0001  0.0012
    0.204               -1.000                        0.0001
10  0.204  1.000  0.994  1.000  0.612  0.920  0.0000  0.0002
11  0.204  1.000  0.996  1.000  0.868  0.973  0.0000  0.0004
    0.204               -1.000                        0.0000
12  0.204  1.000  0.997  1.000  0.955  0.991  0.0000  0.0001
    0.204               -1.000                        0.0000
13  0.204  1.000  0.998  1.000  0.985  0.997  0.0000  0.0000
    0.204               -0.990                        0.0000
14  0.204  1.000  0.999  1.000  0.995  0.999  0.0000  0.0000
15  0.204  1.000  0.999  1.000  0.998  1.000  0.0000  0.0000
        Minimum was achieved after 15 iterations.

     Final Configuration (Stress= 0.203946 Formula 1)

                          D_1_1
                O_1   1.394282445
                O_2  -0.663840798
                O_3   0.746266448
                O_4  -0.042636844
                O_5  -1.434071249

           Group Count      Stress Regression

               1    25  0.20394609 Descending
```

2. Unsymmetric Matrix of Similarity Measures: Regression: Third Order Polynomial

```
simi = [ 9.30  0.57  6.10  2.00  1.50 ,
         1.00  8.50  5.50  3.30  4.00 ,
         2.10  5.30  7.30  3.90  1.10 ,
         5.00  6.20  1.60  8.90  2.80 ,
         1.30  4.10  1.00  2.50  8.80 ];

print "KYST MDS without initial values: Polynomial Degree=3";
optn = [ "data"       "simi" ,
         "unsym"             ,
         "regres"     "poly" ,
```

```
          "poldeg"          3 ,
          "pre-it"          3 ,
          "coord"     "rotat" ,
          "dimmax"          3 ,
          "dimmin"          1 ,
          "phis"              ,
          "norm"             ];
  < gof,conf,resi > = mds("kyst",simi,optn);

               *******************************
               Metric Multidimensional Scaling
               *******************************

             Similarity Input Data (Unsymmetric)
  Number Rows of Matrix . . . . . .          5
  Maximum Dimension . . . . . .              3
  Minimum Dimension . . . . . .              1
  Amount Input Information. . . . . .       25
  Number of Parameters. . . . . .           15
  Degrees of Freedom. . . . . .             10
  Stress Formula. . . . . .                  1
  Diagonal  . . . . . .                 Included
  Polynomial Regression Degree. . . . . .    3
  Maximum Number Iterations . . . . . .   5000
  TORSCA Preiterations. . . . . .            3

          ****************************************
          Compute Kruskal-Young-Shepard-Torgerson KYST
          ****************************************
         There are 25 data values, split into 1 lists.

Torgerson-Young Quasi-Nonmetric Procedure for Initial Configuration

                     Iteration      Stress

                             0  0.022638560
                             1  0.002207875
                             2  0.000459813
                             3  0.000153818
 [note] file tmds1.inp, line 42: Maximum number of pre-iterations 3
           reached.
The best initial configuration by Torgerson-Young quasi-nonmetric
method has Stress=0.000153818 Formula 1

              History of Iteration for 3 Dimensions
```

```
                    ************************************

 Iter STRESS   SRAT SRATAV CAGRCL  COSAV  ACSAV    SFGR    STEP

    0  0.169  0.800  0.800  0.000  0.000  0.000  0.0069  0.0293
    1  0.163  0.968  0.852  0.974  0.643  0.643  0.0056  0.0776
    2  0.153  0.939  0.880  0.738  0.706  0.706  0.0034  0.2093
    3  0.155  1.008  0.921 -0.703 -0.224  0.704  0.0057  0.0997
    4  0.147  0.948  0.930 -0.463 -0.382  0.545  0.0022  0.0464
    5  0.146  0.998  0.952 -0.837 -0.682  0.738  0.0021  0.0153
    6  0.146  0.995  0.966  0.934  0.384  0.867  0.0006  0.0152
    7  0.146  1.000  0.977 -0.897 -0.462  0.887  0.0008  0.0048
    8  0.146  0.999  0.985  0.983  0.492  0.950  0.0003  0.0054
    9  0.146  1.000  0.990 -0.819 -0.374  0.864  0.0002  0.0018
   10  0.146  1.000  0.993  0.401  0.138  0.559  0.0000  0.0012
   11  0.146  1.000  0.995  0.574  0.426  0.569  0.0000  0.0016
   12  0.146  1.000  0.997 -0.448 -0.151  0.489  0.0001  0.0008
   13  0.146  1.000  0.998 -0.553 -0.416  0.531  0.0000  0.0003
   14  0.146  1.000  0.999 -0.631 -0.558  0.597  0.0000  0.0001
   15  0.146  1.000  0.999  0.901  0.405  0.798  0.0000  0.0001
          Minimum was achieved after 15 iterations.

        Final Configuration (Stress= 0.145597 Formula 1)

                      D_3_1        D_3_2        D_3_3
          O_1   0.966987925 -0.512572882 -0.077858169
          O_2  -0.609811159  0.628542479  0.045316107
          O_3   0.480826435  0.706684142 -0.399330276
          O_4  -0.075197068 -0.128523217  0.908024094
          O_5  -0.762806134 -0.694130522 -0.476151756
```

The regression coefficients start with the intercept and it is of no surprise that the coefficient of the order 3 term is negative indicating an descending behavior:

```
          Group Count      Stress Regression Coefficients

              1    25  0.14559717   2.003  -0.206   0.030  -0.004

               History of Iteration for 2 Dimensions
               ***********************************

 Iter STRESS   SRAT SRATAV CAGRCL  COSAV  ACSAV    SFGR    STEP

    0  0.209  0.800  0.800  0.000  0.000  0.000  0.0054  0.0281
    1  0.206  0.986  0.858  0.929  0.613  0.613  0.0036  0.0714
```

```
 2  0.203  0.983  0.898  0.149  0.307  0.307  0.0028  0.1094
 3  0.205  1.012  0.934 -0.635 -0.315  0.524  0.0073  0.0547
 4  0.200  0.973  0.947  0.532  0.244  0.529  0.0015  0.0535
 5  0.200  1.002  0.965 -0.631 -0.333  0.596  0.0035  0.0232
 6  0.199  0.994  0.975  0.775  0.398  0.714  0.0010  0.0260
 7  0.199  1.000  0.983 -0.610 -0.267  0.645  0.0014  0.0109
 8  0.199  0.999  0.988  0.742  0.399  0.709  0.0005  0.0119
 9  0.199  1.000  0.992 -0.035  0.113  0.264  0.0005  0.0098
10  0.199  1.000  0.995 -0.483 -0.281  0.409  0.0005  0.0048
11  0.198  1.000  0.996  0.293  0.098  0.332  0.0002  0.0036
12  0.198  1.000  0.998  0.805  0.564  0.644  0.0002  0.0059
13  0.198  1.000  0.998  0.068  0.237  0.264  0.0002  0.0064
14  0.198  1.000  0.999 -0.851 -0.481  0.652  0.0005  0.0022
15  0.198  1.000  0.999  0.985  0.486  0.872  0.0003  0.0025
       Minimum was achieved after 15 iterations.

     Final Configuration (Stress= 0.198448 Formula 1)

                        D_2_1        D_2_2
               O_1   1.144281569 -0.497908299
               O_2  -0.620555127  0.658039311
               O_3   0.572111824  0.801728943
               O_4  -0.004133515 -0.487920895
               O_5  -1.091704751 -0.473939059

     Group Count      Stress Regression Coefficients

         1    25  0.19844843   2.754  -0.894   0.176  -0.012

          History of Iteration for 1 Dimensions
          *************************************

Iter STRESS   SRAT SRATAV CAGRCL  COSAV  ACSAV    SFGR    STEP

   0  0.332  0.800  0.800  0.000  0.000  0.000  0.0078  0.0647
      0.330                -0.967                        0.0065
   1  0.332  0.998  0.861  1.000  0.660  0.660  0.0067  0.0175
   2  0.330  0.996  0.904  0.998  0.883  0.883  0.0037  0.0593
      0.331                -0.989                        0.0059
   3  0.330  0.999  0.935  1.000  0.960  0.960  0.0027  0.0210
   4  0.330  0.999  0.956 -0.929 -0.287  0.939  0.0009  0.0076
      0.330                -0.998                        0.0008
   5  0.330  1.000  0.970  1.000  0.563  0.979  0.0007  0.0010
   6  0.330  1.000  0.980  1.000  0.851  0.993  0.0005  0.0024
   7  0.330  1.000  0.987  0.995  0.946  0.994  0.0001  0.0071
```

```
    0.330                -0.999                          0.0007
 8  0.330  1.000  0.991 -0.430  0.038  0.622  0.0000  0.0004
    0.330                -0.992                          0.0000
 9  0.330  1.000  0.994  0.466  0.321  0.519  0.0000  0.0000
    0.330                -0.995                          0.0000
10  0.330  1.000  0.996  0.953  0.738  0.805  0.0000  0.0000
    0.330                -1.000                          0.0000
11  0.330  1.000  0.997  0.998  0.910  0.933  0.0000  0.0000
    0.330                -1.000                          0.0000
12  0.330  1.000  0.998  1.000  0.969  0.977  0.0000  0.0000
    0.330                -1.000                          0.0000
13  0.330  1.000  0.999  1.000  0.990  0.992  0.0000  0.0000
    0.330                -1.000                          0.0000
14  0.330  1.000  0.999  1.000  0.996  0.997  0.0000  0.0000
        Minimum was achieved after 14 iterations.

     Final Configuration (Stress= 0.329676 Formula 1)

                              D_1_1
                    O_1   1.452973688
                    O_2  -0.710018727
                    O_3   0.707880939
                    O_4  -0.080754068
                    O_5  -1.370081832

     Group Count      Stress Regression Coefficients

         1    25  0.32967640   3.139  -1.212   0.223  -0.014
```

- Test Examples from MULTISCALE Manual
  1. Example HANGUL by Yoshio Takane: 3-way Rectangular Data: m=3, n=14

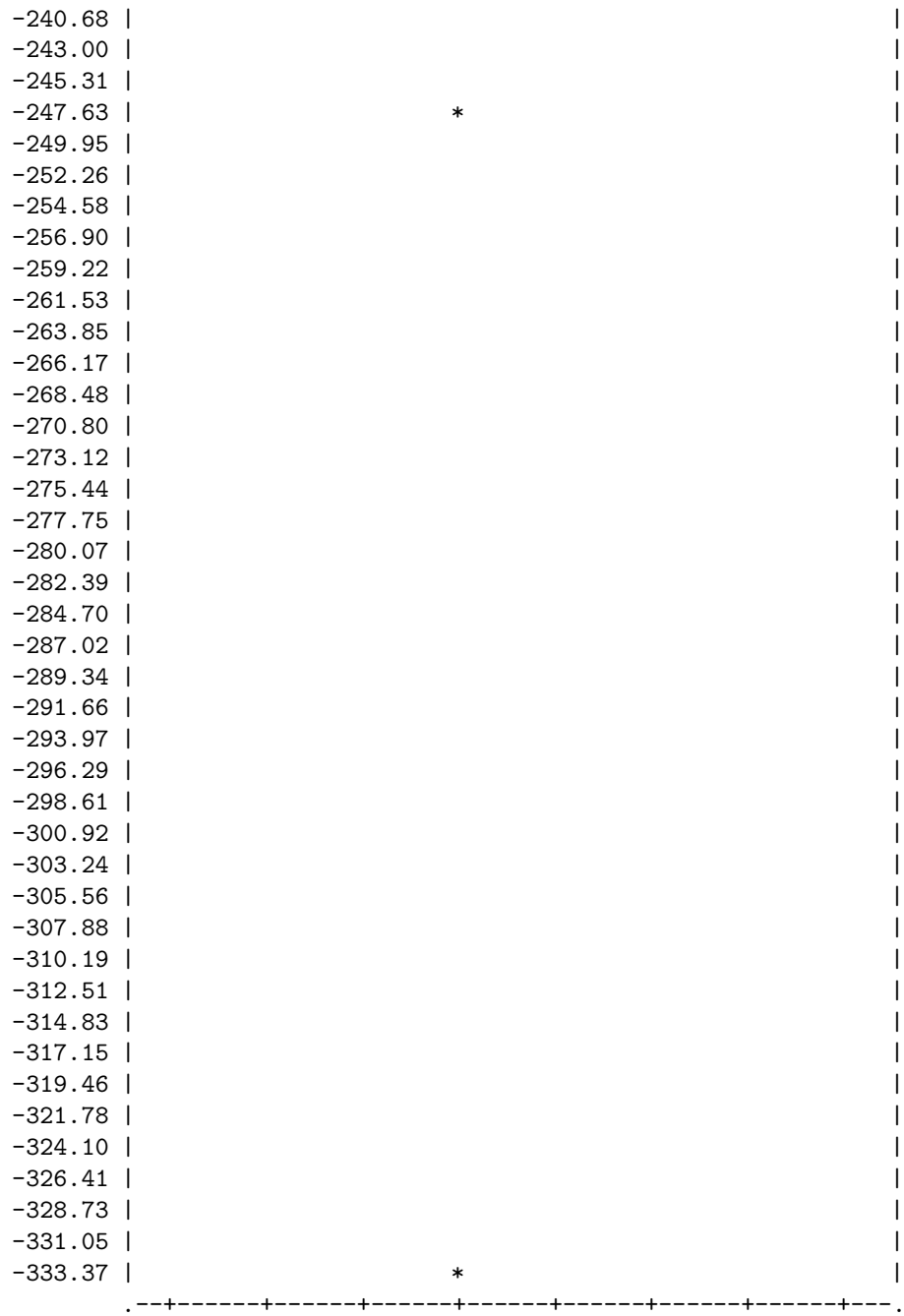
```
print "HANGUL: 3-way Rectangular Data: Nsub=3, Nobs=14";
n = 14; nw = 3; ndim = 3;
order = [ 24 31  8 42 21 91 25 55 85 15 50 51 27 30 39 26  2 88 87 29 68
          53 49 72 69 67 22 75 57 44 16 19 36 48 10 78 33  7 35  4 61 46
          83 38 70 54 77 74 60 28  3 11 80 64 66 40 43 52 79 56 76 47 71
          90 14 32  6 12 81 23 65  9 63 45 41 73 37 59 58 89 84  5 20 62
          18 13 34 17  1 82 86 ];
dis1 = [ 4 6 3 4 3 1 3 3 2 6 6 7 3 4 6 4 1 2 3 6 6 7 6 1 4 6 6 4 7 4 6 7 6 1
         1 4 4 2 4 1 1 6 7 4 6 6 4 7 4 6 6 7 6 3 6 1 4 6 7 7 8 6 2 4 6 6 7 7
         3 6 4 2 6 6 7 5 6 3 1 4 7 6 8 6 7 7 4 4 3 ];
dis2 = [ 2 3 1 6 7 1 6 3 5 5 6 5 2 3 6 6 1 2 4 6 3 7 8 3 6 5 6 7 5 6 6 7 6 2
         1 6 4 2 5 7 3 5 6 7 7 6 6 7 8 4 7 8 5 4 7 2 4 8 4 3 7 7 4 3 5 4 6 7
         2 7 7 2 7 5 6 6 3 6 2 5 6 2 7 3 7 6 5 8 4 ];
```

```
dis3 = [ 3 5 2 9 4 1 5 4 8 9 8 8 5 9 8 7 1 6 6 6 8 8 9 2 5 5 6 8 9 8 7 8 4 1
         1 9 7 6 8 2 2 7 8 9 6 6 3 8 7 6 5 6 4 3 8 7 6 6 6 7 9 8 2 8 8 6 7 6
         6 7 6 1 9 7 7 9 1 7 2 9 7 7 8 2 6 9 6 9 3 ];
dis1[order] = dis1; dis2[order] = dis2; dis3[order] = dis3;

/* get tensor diss[nw,n,n] */
real diss[nw,n,n];
diss[1,.,.] = vec2tri(dis1,n,"ssym"); diss[2,.,.] = vec2tri(dis2,n,"ssym");
diss[3,.,.] = vec2tri(dis3,n,"ssym");
print "Tensor diss=",diss;

rnam = [" GA NA DA LA MA BA SA AA JA CHA KA TA PA HA "];
wnam = [" NAOMI ISABELLE OLEX "];
list tnam[3];
tnam[1] = wnam; tnam[2] = tnam[3] = rnam;
diss = dimname(diss,tnam);
print "Tensor diss=",diss;

 print "HANGUL: Symmetric: nsub=3, nobs=14, ndim=3";
 optn = [ "data"       "diss" ,
          "symm"              ,
          "model"       "dia" ,
          "regres"    "power" ,
          "ndim"            3 ,
          "ptab"              ,
          "pdis"              ,
          "pres"              ,
          "phis"              ];
 options LS=80;
 < gof,conf,resi,wgt > = mds("mult",diss,optn);

                ****************************
                3-Way Multidimensional Scaling
                ****************************

             Dissimilarity Input Data (Symmetric)
  Number Rows of Matrix . . . . . .          14
  Dimension . . . . . .           3
  Number Subjects / Replications. . . . . .          3
  Amount Input Information. . . . . .         273
  Number of Parameters. . . . . .          51
  Degrees of Freedom. . . . . .         222
  Maximum Number Iterations . . . . . .          50
  Convergence Criterion for Main Iterations . . .        0.05
  Number of Constraint Equations. . . . . .           0
```

```
Individual Metric Matrix. . . . . .   DIAGONAL
Estimated Data Transformation Type. . . . . .      POWER
Stimulus Variance Type. . . . . .   CONSTANT
Subject Variance Type . . . . . .    SUBWISE
Assumed Error Distribution. . . . . .  LogNormal
Maximum Number of Configuration Iterations. . . .          1
Convergence Criterion for Configuration . . . .  0.000366367
Maximum Number of Metric Matrix Iterations. . . .          1
Convergence Criterion for Metric Matrix . . . .   0.00109951
Standard Error Weight Estimates . . . . . .   Post-Hoc

          ************************************
          Compute Metric MULTISCALE by J. Ramsay
          ************************************

            Zero Data: 0   Nonzero Data: 546

               Initial Configuration Matrix

                DIM_1        DIM_2        DIM_3
      GA    1.693392140  3.440379391 -0.058330375
      NA    3.821119948 -0.865510991 -1.535791450
      DA    1.242735081  2.824937623  2.057309005
      LA    2.926946574 -0.817396710  0.605060613
      MA    3.431472988 -0.370735784 -2.321494491
      BA   -0.457316347 -1.568108691  2.953564288
      SA   -2.857053380 -0.876447666 -2.282018660
      AA    1.546868870 -2.013375035  1.268114706
      JA   -0.740375144 -0.671116637 -2.832730182
      CHA  -3.171367453 -0.345144311 -2.079653150
      KA   -2.081242024  4.070858224 -0.393830538
      TA   -2.381454586  0.530835880  1.677753277
      PA   -0.755927749 -1.184727728  2.382583253
      HA   -2.217798916 -2.154447564  0.559463706

 MainIter=0 ErrSS=34.7658 LogLik=-332.275 Std.Err=0.504672
                       Grad=3.37034

******************** Main Iteration 1 ********************


                  ************************
                  Configuration Iterations
                  ************************
```

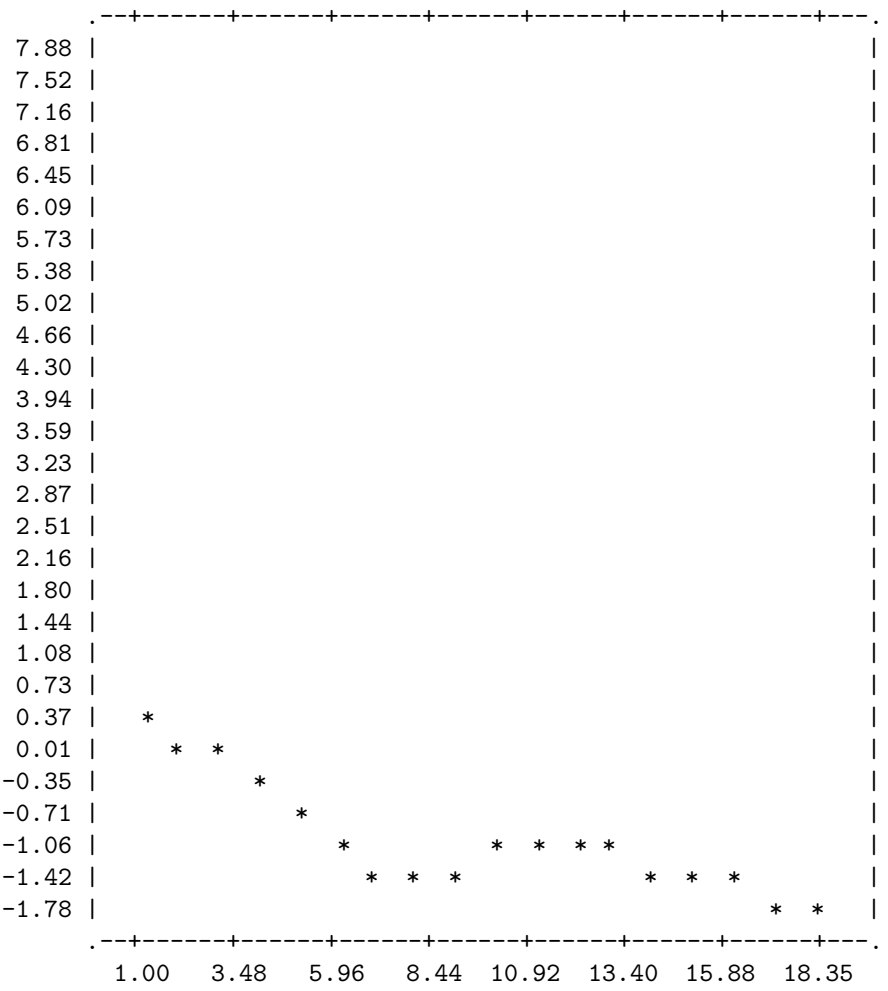
```
              Iter  ErrorSS  Penalty    Grad

            0 34.76577380 0.000000000 3.242322213

   Line Search Convergence achieved after 4 Iterations.


            0 20.19175731 0.000000000 0.808229728

                 ************************
                 Weight Matrix Iterations
                 ************************

             Subj  Iter     ErrorSS     GradLgn
   Line Search Convergence achieved after 3 Iterations.
                1     0 5.098514191 0.018974946
   Line Search Convergence achieved after 3 Iterations.
                2     0 8.235164576 0.028990435
   Line Search Convergence achieved after 4 Iterations.
                3     0 6.210575504 0.067967736

  MainIter=1 ErrSS=18.5833 LogLik=-247.913 Std.Err=0.368973
                       Grad=0.781247


******************** Main Iteration 17 ********************


                 ************************
                 Configuration Iterations
                 ************************

              Iter  ErrorSS  Penalty    Grad

            0 16.07777032 0.000000000 0.028864305

   Line Search Convergence achieved after 3 Iterations.


            0 16.07443345 0.000000000 0.016972150
        Configuration matrix convergence achieved.

                 ************************
                 Weight Matrix Iterations
                 ************************
```

```
                Subj  Iter     ErrorSS     GradLgn


  MainIter=17 ErrSS=16.0703 LogLik=-222.141 Std.Err=0.34312
                      Grad=0.0169858
       Main Convergence achieved after 17 Iterations.

              *********************************
              Post-Mortem Display of Iterations
              *********************************

            Iter   LogLikel.    ErrorSSQ Grad_Length

              0 -332.275034  34.7657738  3.37034133
              1 -247.912682  18.5832600  0.78124681
              2 -237.995998  17.2037229  0.68795646
              3 -228.132661  16.1328279  0.43250699
              4 -224.694652  16.0185771  0.16987826
              5 -224.111344  16.1412531  0.07203409
              6 -223.947027  16.2305851  0.04191853
              7 -223.834008  16.2993201  0.03925094
              8 -223.718201  16.3456821  0.04228601
              9 -223.562014  16.3497568  0.05768708
             10 -223.325721  16.3119669  0.06698664
             11 -222.986955  16.2402661  0.07156129
             12 -222.696816  16.1705122  0.06266364
             13 -222.459718  16.1037396  0.04617941
             14 -222.320727  16.0801308  0.03924391
             15 -222.224806  16.0631223  0.02864440
             16 -222.170350  16.0777703  0.02346187
             17 -222.140745  16.0703107  0.01698578

          Log Likelihood Versus Iterations (Dimension=3)
          **********************************************

         .--+------+------+------+------+------+------+------+---.
  -222.14 |                            MMM                       |
  -224.46 |                          MMM                         |
  -226.78 |                                                      |
  -229.09 |                         *                            |
  -231.41 |                                                      |
  -233.73 |                                                      |
  -236.04 |                                                      |
  -238.36 |                         *                            |
```



Plot of LOG of Gradient Length vs. Iterations (Dim=3)

****



****
ML Results for 3 Dimensions
****

Log Likelihood.	-222.141
AIC Statistic	544.281
BIC Statistic	724.755
Unbiased Standard Error Estimate.	0.379642
Number of Parameters.	50
Number of Degrees of Freedom for Error.	223
Maximum Likelihood Standard Error Estimate. . .	0.34312

****
ML Results for Zero Dimensions
****

Log Likelihood.	-397.547
AIC Statistic	801.093
BIC Statistic	811.922
Unbiased Standard Error Estimate.	0.572895
Number of Parameters.	3
Number of Degrees of Freedom for Error.	270
Chi Square for Comparing two Fits	350.812
Number of Degrees of Freedom for Chi Square . . .	47
Probability of Exceeding this Chi Square. . . .	0

Final Configuration Matrix

	DIM_1	DIM_2	DIM_3
GA	-5.833769549	42.95111349	17.49054187
NA	33.71451814	23.01910644	-46.62252275
DA	-6.972180399	32.30519044	26.21958755
LA	74.17208531	4.518121476	3.582671946
MA	40.00171750	38.78290758	-26.40205602
BA	14.43745091	-27.51251267	22.23081108
SA	-61.36680439	-19.85491655	-14.00618563
AA	32.53634008	-18.94987224	-12.58251102
JA	-34.72490670	-2.986183306	-33.73231834
CHA	-35.02008594	-20.98091483	-21.01765981
KA	-26.82689028	31.78583605	21.54301983
TA	-14.13338676	-18.90323856	28.52173339
PA	-6.551859311	-31.25637806	24.62244759
HA	-3.432228613	-32.91825927	10.15244032

The decimal point has been moved 1 places to the right.

Results in Polar Coordinate Form

	1	2	3
GA	46.741310	98.000000	108.00000
NA	61.969409	34.000000	-54.000000
DA	42.186531	102.00000	105.00000
LA	74.395882	3.0000000	3.0000000
MA	61.654845	44.000000	-33.000000
BA	38.204546	-62.000000	57.000000
SA	66.002088	198.00000	193.00000
AA	39.699253	-30.000000	-21.000000
JA	48.503667	185.00000	224.00000

CHA	45.916742	211.00000	211.00000
KA	46.841468	130.00000	141.00000
TA	37.021269	233.00000	116.00000
PA	40.325587	258.00000	105.00000
HA	34.618839	264.00000	109.00000

First Value is Distance from the Origin
Other Values are Angles in Degrees from First Dimension

****
Standard Deviations and Variances for each Dimension
****

Dim	Std.Dev.	Variance	Percent
1	34.60001437	1197.160994	47
2	27.16411058	737.8889034	29
3	24.33720052	592.2993292	23

Interpoint Distances

	GA	NA	DA	LA
GA	0	77	13	89
NA	77	0	83	67
DA	13	83	0	88
LA	89	67	88	0
MA	63	26	70	56
BA	73	87	63	70
SA	89	109	85	138
AA	78	54	75	50
JA	74	74	74	115
CHA	80	85	76	114
KA	24	91	20	106
TA	63	98	51	94
PA	74	98	63	90
HA	76	87	67	86

.....

Final Weight Matrix

	DIM_1	DIM_2	DIM_3
NAOMI	0.412642095	1.155887660	1.222149835
ISABELLE	1.218159780	1.158094466	0.418215205
OLEX	1.160933994	0.567936658	1.153117607

Results in Polar Coordinate Form

	1	2	3
NAOMI	1.7320508	70.000000	45.000000
ISABELLE	1.7320508	44.000000	43.000000
OLEX	1.7320508	26.000000	0.0000000

First Value is Distance from the Origin
Other Values are Angles in Degrees from First Dimension

Post-Hoc Standard Error Weight Estimates

	Input Order			Size Order	
1	GA	0.00000000	1	GA	0.00000000
2	NA	0.61133448	4	LA	0.16585465
3	DA	2.29960410	5	MA	0.25344875
4	LA	0.16585465	2	NA	0.61133448
5	MA	0.25344875	13	PA	0.64146629
6	BA	1.26767980	11	KA	0.81903743
7	SA	1.66273222	14	HA	0.87274233
8	AA	1.25955433	9	JA	1.06895864
9	JA	1.06895864	8	AA	1.25955433
10	CHA	1.49133855	6	BA	1.26767980
11	KA	0.81903743	10	CHA	1.49133855
12	TA	2.00985566	7	SA	1.66273222
13	PA	0.64146629	12	TA	2.00985566
14	HA	0.87274233	3	DA	2.29960410

Final Regression Coefficient and Exponent Estimates

	Subject	Regression Coefficient	Exponent	Reciprocal		Subject	Exponent by Size
1	NAOMI	1.40680746	0.93303742	1.07176838	1	NAOMI	0.93303742
2	ISABELLE	0.72030562	1.31213620	0.76211601	3	OLEX	0.99164164
3	OLEX	0.98684407	0.99164164	1.00842881	2	ISABELLE	1.31213620

Mean= 1.07894 Standard Deviation= 0.20407

Within-Subj. Std.Error Estimates and Multiple Correlations

Standard Error	Multiple Correlation

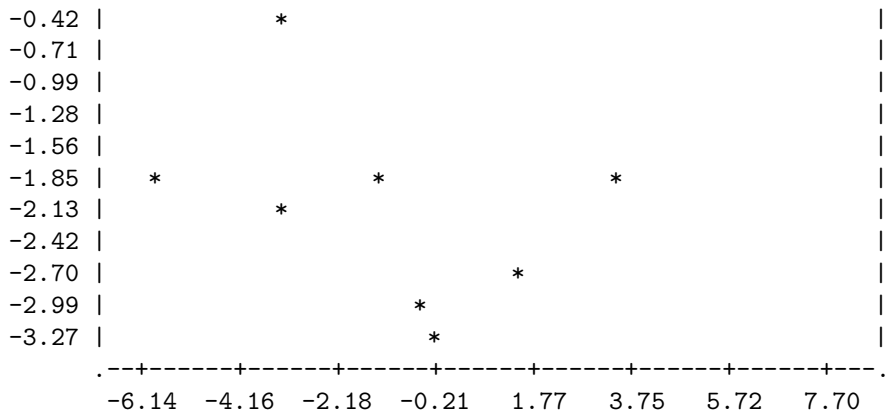
Subj	Unordered Subj		Ordered	Subj	Unordered Subj		Ordered
1	0.22	1	0.22	1	0.91	2	0.67
2	0.45	3	0.32	2	0.67	3	0.75
3	0.32	2	0.45	3	0.75	1	0.91

```
************************************************
Asympt. Standard Errors of Estimate of Coordinates
************************************************

   1          GA    -0.5834     4.2951     1.7491
                     0.7183     0.8379     0.6427

  Correlation Matrix 1 for Coordinate Estimates

               DIM_1        DIM_2        DIM_3
DIM_1   1.000000000 -0.024978736  0.440441259
DIM_2  -0.024978736  1.000000000 -0.092418119
DIM_3   0.440441259 -0.092418119  1.000000000

   2          NA     3.3715     2.3019    -4.6623
                     0.4957     0.7825     0.4466

  Correlation Matrix 2 for Coordinate Estimates

               DIM_1        DIM_2        DIM_3
DIM_1   1.000000000  0.033762208 -0.100446586
DIM_2   0.033762208  1.000000000 -0.455694562
DIM_3  -0.100446586 -0.455694562  1.000000000

   3          DA    -0.6972     3.2305     2.6220
                     0.4882     0.5092     0.6462

  Correlation Matrix 3 for Coordinate Estimates

               DIM_1        DIM_2        DIM_3
DIM_1   1.000000000  0.133052899  0.175968019
DIM_2   0.133052899  1.000000000  0.077459569
DIM_3   0.175968019  0.077459569  1.000000000

   4          LA     7.4172     0.4518     0.3583
                     0.5384     0.5615     0.5177

  Correlation Matrix 4 for Coordinate Estimates

               DIM_1        DIM_2        DIM_3
```

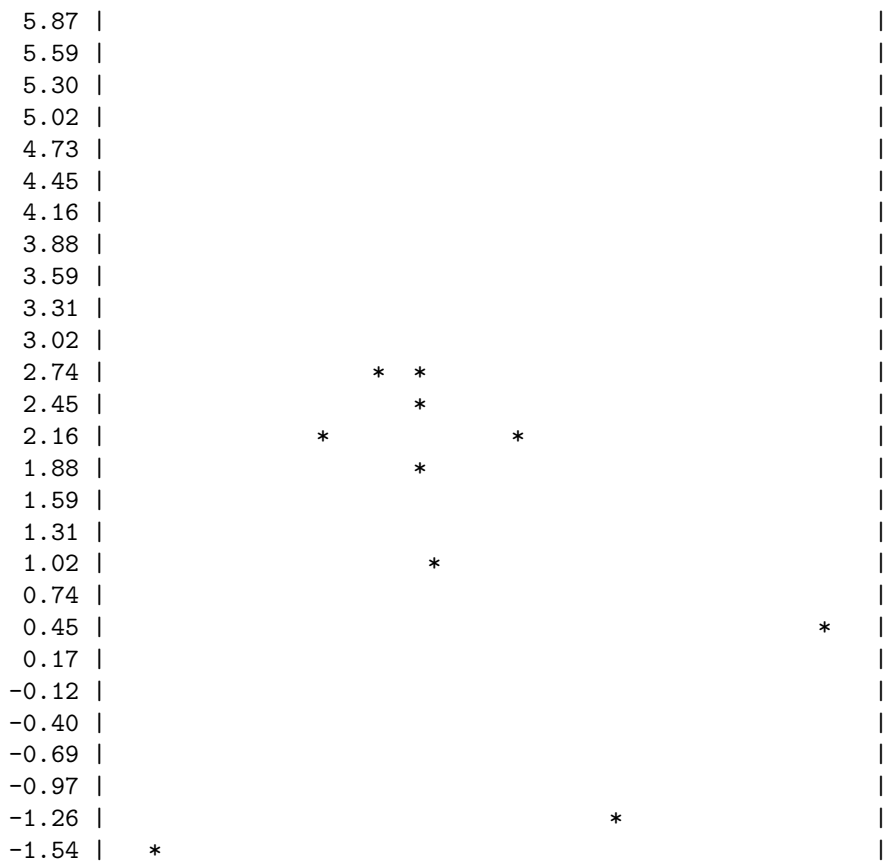
```
DIM_1   1.000000000 -0.485018869 -0.046880810
DIM_2  -0.485018869  1.000000000 -0.209008714
DIM_3  -0.046880810 -0.209008714  1.000000000

  5          MA     4.0002     3.8783    -2.6402
                    0.5040     0.5151     0.4546

Correlation Matrix 5 for Coordinate Estimates

              DIM_1        DIM_2        DIM_3
DIM_1   1.000000000  0.491034736 -0.057189755
DIM_2   0.491034736  1.000000000 -0.071323172
DIM_3  -0.057189755 -0.071323172  1.000000000

  6          BA     1.4437    -2.7513     2.2231
                    0.9316     0.5191     0.8535

Correlation Matrix 6 for Coordinate Estimates

              DIM_1        DIM_2        DIM_3
DIM_1   1.000000000 -0.588478790  0.025758511
DIM_2  -0.588478790  1.000000000 -0.107072269
DIM_3   0.025758511 -0.107072269  1.000000000

  7          SA    -6.1367    -1.9855    -1.4006
                    0.7397     0.5875     0.9220

Correlation Matrix 7 for Coordinate Estimates

              DIM_1        DIM_2        DIM_3
DIM_1   1.000000000 -0.304544151 -0.114603685
DIM_2  -0.304544151  1.000000000 -0.361201975
DIM_3  -0.114603685 -0.361201975  1.000000000

  8          AA     3.2536    -1.8950    -1.2583
                    0.6351     0.7670     0.6686

Correlation Matrix 8 for Coordinate Estimates

              DIM_1        DIM_2        DIM_3
DIM_1   1.000000000 -0.063057167 -0.127590439
DIM_2  -0.063057167  1.000000000  0.671252254
DIM_3  -0.127590439  0.671252254  1.000000000

  9          JA    -3.4725    -0.2986    -3.3732
```

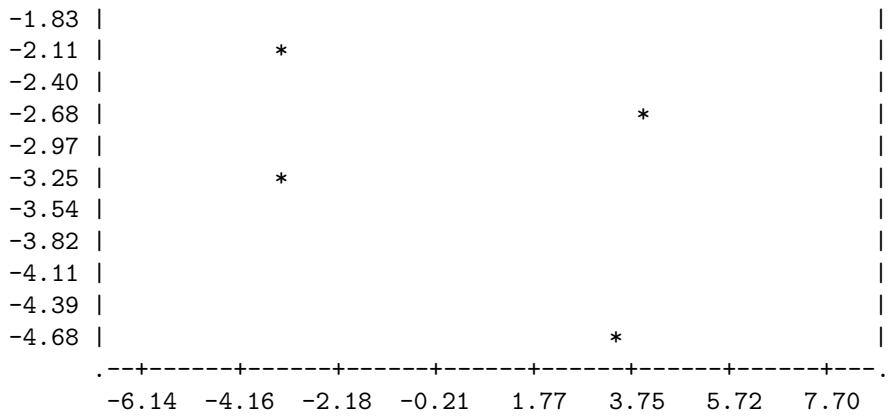
```
                    0.5475     0.5658     0.5126

Correlation Matrix 9 for Coordinate Estimates

              DIM_1        DIM_2        DIM_3
DIM_1   1.000000000  0.513133620  0.456933319
DIM_2   0.513133620  1.000000000  0.781066209
DIM_3   0.456933319  0.781066209  1.000000000

 10         CHA    -3.5020    -2.0981    -2.1018
                    0.4702     0.7437     0.5956

Correlation Matrix 10 for Coordinate Estimates

              DIM_1        DIM_2        DIM_3
DIM_1   1.000000000 -0.264506346 -0.196169208
DIM_2  -0.264506346  1.000000000  0.319587558
DIM_3  -0.196169208  0.319587558  1.000000000

 11          KA    -2.6827     3.1786     2.1543
                    0.6329     0.8756     0.7605

Correlation Matrix 11 for Coordinate Estimates

              DIM_1        DIM_2        DIM_3
DIM_1   1.000000000 -0.080332951  0.401870991
DIM_2  -0.080332951  1.000000000  0.213916668
DIM_3   0.401870991  0.213916668  1.000000000

 12          TA    -1.4133    -1.8903     2.8522
                    0.5556     0.6714     0.5545

Correlation Matrix 12 for Coordinate Estimates

              DIM_1        DIM_2        DIM_3
DIM_1   1.000000000  0.139738633  0.404038865
DIM_2   0.139738633  1.000000000 -0.109471051
DIM_3   0.404038865 -0.109471051  1.000000000

 13          PA    -0.6552    -3.1256     2.4622
                    0.5078     0.5252     0.7352

Correlation Matrix 13 for Coordinate Estimates

              DIM_1        DIM_2        DIM_3
```

```
DIM_1   1.000000000  0.336710058 -0.014465145
DIM_2   0.336710058  1.000000000 -0.240615577
DIM_3  -0.014465145 -0.240615577  1.000000000

 14          HA    -0.3432    -3.2918     1.0152
                    0.4934     0.5487     0.5469

Correlation Matrix 14 for Coordinate Estimates

              DIM_1        DIM_2        DIM_3
DIM_1   1.000000000  0.505301215  0.531299385
DIM_2   0.505301215  1.000000000  0.877935810
DIM_3   0.531299385  0.877935810  1.000000000

Configuration: Dimension 2 (Y-Axis) vs. Dimension 1 (X-Axis)
**********************************************************

     .--+------+------+------+------+------+------+------+---.
 7.28 |                                                       |
 6.99 |                                                       |
 6.71 |                                                       |
 6.42 |                                                       |
 6.14 |                                                       |
 5.85 |                                                       |
 5.57 |                                                       |
 5.28 |                                                       |
 5.00 |                                                       |
 4.71 |                                                       |
 4.43 |                     *                                 |
 4.14 |                                                       |
 3.86 |                                    *                  |
 3.57 |                                                       |
 3.29 |               *     *                                 |
 3.00 |                                                       |
 2.72 |                                                       |
 2.43 |                                  *                    |
 2.15 |                                                       |
 1.86 |                                                       |
 1.58 |                                                       |
 1.29 |                                                       |
 1.00 |                                                       |
 0.72 |                                                       |
 0.43 |                                                   *   |
 0.15 |                                                       |
-0.14 |                                                       |
```

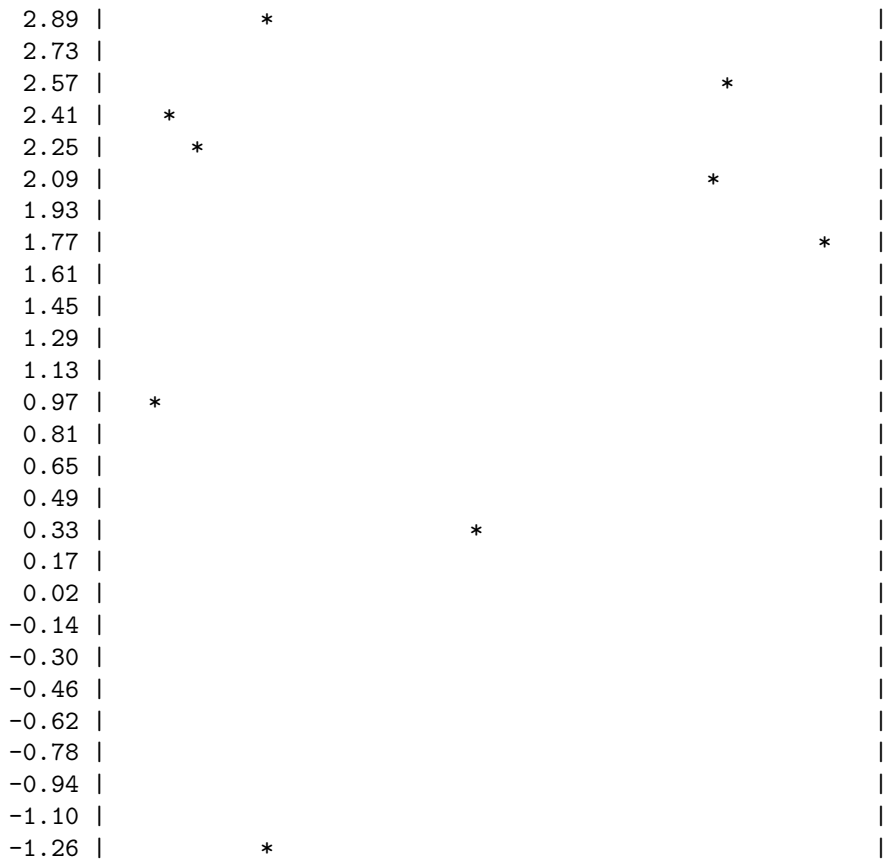


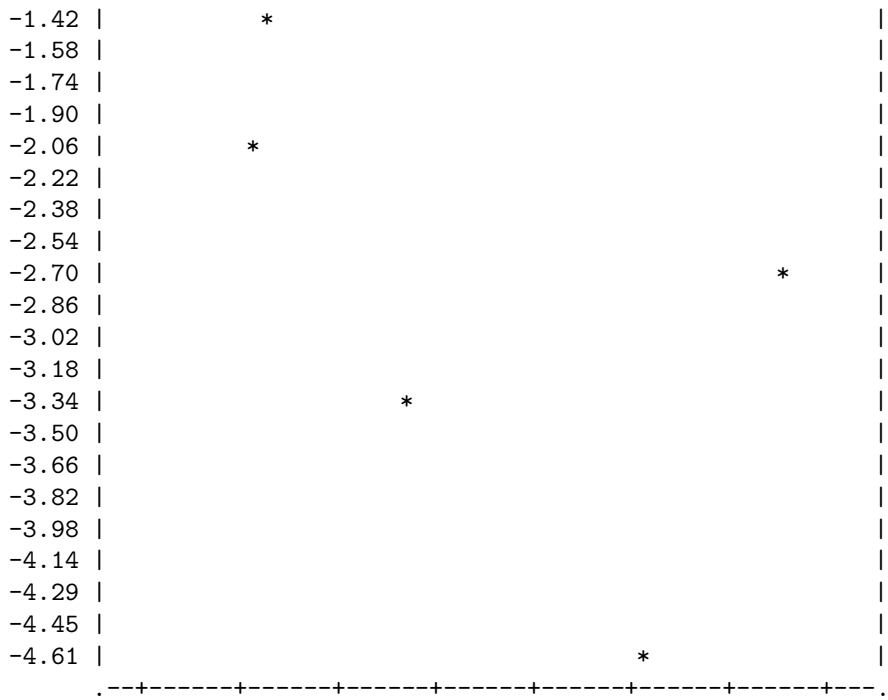
Configuration: Dimension 3 (Y-Axis) vs. Dimension 1 (X-Axis)
****





Configuration: Dimension 3 (Y-Axis) vs. Dimension 2 (X-Axis)
****





Transformed Observations: Subject=1

	GA	NA	DA	LA
GA	0.000000000	5.128367782	1.406807460	5.128367782
NA	5.128367782	0.000000000	7.486501175	5.128367782
DA	1.406807460	7.486501175	0.000000000	7.486501175
LA	5.128367782	5.128367782	7.486501175	0.000000000
MA	5.128367782	3.921088520	7.486501175	3.921088520
BA	7.486501175	8.644557291	7.486501175	5.128367782
SA	7.486501175	8.644557291	9.791549267	8.644557291
AA	7.486501175	3.921088520	5.128367782	3.921088520
JA	7.486501175	5.128367782	7.486501175	7.486501175
CHA	8.644557291	8.644557291	7.486501175	7.486501175
KA	1.406807460	9.791549267	1.406807460	7.486501175
TA	8.644557291	8.644557291	7.486501175	6.315385005
PA	7.486501175	7.486501175	5.128367782	5.128367782
HA	7.486501175	8.644557291	8.644557291	5.128367782

Smallest Value: 1.40681 Largest Value: 9.79155

Interpoint Distances: Subject=1

```
              GA           NA           DA           LA
GA    0.000000000  7.158159131  1.370558004  6.191635096
NA    7.158159131  0.000000000  7.795151562  5.891682281
DA    1.370558004  7.795151562  0.000000000  5.951115483
LA    6.191635096  5.891682281  5.951115483  0.000000000
MA    5.206060788  2.591533808  6.026122181  4.956083155
BA    7.045076335  8.625782950  6.027053099  5.077400153
SA    7.674641926  7.734798475  7.282667820  8.502045660
AA    7.166111207  5.372933335  6.794128139  3.738176556
JA    7.077124791  4.939050764  7.170939395  7.462352230
CHA   7.589068483  6.458338546  7.277167683  7.321478936
KA    1.700929113  7.801900058  1.259435926  6.758443997
TA    6.201400534  9.087634037  5.057124107  6.209752037
PA    7.330885611  9.269960481  6.250776957  6.274938090
HA    7.496727933  8.242553984  6.617793875  5.896113597
.....................................................
    Smallest Value: 1.25944 Largest Value: 9.26996

            Normalized Residuals: Subject=1

              MA           BA           SA           AA
GA   -0.067030618  0.270923841 -0.110647805  0.194985653
NA    1.846142233  0.009692450  0.495731186 -1.404289434
DA    0.967375305  0.966686685  1.319666588 -1.253906046
LA   -1.044269497  0.044526847  0.074105732  0.212964481
MA    0.000000000 -0.460180256  0.137649537 -0.595626465
BA   -0.460180256  0.000000000 -1.756529498  0.173533385
SA    0.137649537 -1.756529498  0.000000000 -0.326683485
AA   -0.595626465  0.173533385 -0.326683485  0.000000000
JA   -0.741271493  1.074014716 -0.535849169  2.020291872
CHA   0.073331919 -0.136792760 -0.858941556  1.037428300
KA    0.787188369  1.400725162 -1.085591432  0.943633307
TA    0.049187159 -1.533094294  1.696078236  0.132744744
PA   -0.834288953  0.320711097  1.638671557  0.514433155
HA   -0.490682645 -0.832249088 -2.180979586  0.622017989
.....................................................
    Smallest Value: -2.42579 Largest Value: 2.43163

      Quantiles of Normalized Residuals: Subject=1

              GA           NA           DA           LA
GA    0.000000000 -1.359962536  0.054375453 -0.744341114
NA   -1.359962536  0.000000000 -0.332563025 -0.574739567
DA    0.054375453 -0.332563025  0.000000000  1.027144882
```

```
LA   -0.744341114 -0.574739567  1.027144882  0.000000000
MA   -0.247313727  1.844177266  0.938731970 -0.981979846
BA    0.390753455 -0.191527695  0.897170710 -0.136356872
SA   -0.275495595  0.511531310  1.294223527 -0.054375453
AA    0.275495595 -1.294223527 -1.233639904  0.332563025
JA    0.361503163  0.163876553  0.247313727 -0.163876553
CHA   0.640341274  1.233639904  0.081626883  0.027173016
KA   -0.780818419  0.981979846  0.480696247  0.450314414
TA    1.432159890 -0.361503163  1.712054734 -0.027173016
PA    1.0101e-007 -0.938731970 -0.857103039 -0.897170710
HA   -0.219332424  0.303903190  1.177268879 -0.607212923
.....................................................
    Smallest Value: -2.29533 Largest Value: 2.29533
```

2. Example: MDS of dissimilarity judgements among 10 simple vowels: m=3, n=14

```
n = 10; nw = 4;
order = [ 15 16 24 13 18  7 29 39 17 19 42 14 11 27 36
           1  3 20  5 37 22  9  4 10 35 26 40 33 30 43
          28  2 45 38 23 25 41  6 12  8 44 31 32 34 21 ];
dis1 = [ 4 5 6 5 6 4 6 2 6 5 6 5 5 4 7
         3 2 2 3 2 4 4 3 5 4 5 2 3 6 3
         3 2 5 2 5 6 5 1 2 5 4 5 5 4 4 ];
dis2 = [ 4 4 8 5 4 2 7 1 7 8 6 1 7 2 6
         1 2 1 1 1 3 3 3 6 8 2 4 2 6 3
         1 1 1 5 4 1 3 1 2 3 5 2 8 3 3 ];
dis3 = [ 4 4 8 7 5 6 5 1 5 6 7 5 6 4 5
         2 3 3 2 1 6 5 4 8 3 5 3 4 6 4
         2 2 1 3 5 4 5 2 1 8 5 5 7 7 4 ];
dis4 = [ 5 3 7 3 4 4 2 2 4 6 5 4 3 2 6
         2 2 2 2 1 6 4 2 8 5 4 2 4 7 5
         4 2 3 2 4 2 6 2 5 3 4 3 7 8 4 ];
ratdata = [ 5 5 5 5 5 7 7 7 7 6 5 6 5 6 4 6 4 4 1 1
            4 3 4 5 6 3 6 7 5 5 2 3 3 2 2 6 1 8 1 1 ];
dis1[order] = dis1; dis2[order] = dis2; dis3[order] = dis3;
dis4[order] = dis4;

/* get tensor diss[nw,n,n] */
n = 10; nw = 4;
real diss[nw,n,n];
diss[1,.,.] = vec2tri(dis1,n,"ssym"); diss[2,.,.] = vec2tri(dis2,n,"ssym");
diss[3,.,.] = vec2tri(dis3,n,"ssym"); diss[4,.,.] = vec2tri(dis4,n,"ssym");

rnam = [" A AA E EE O OO U UU MI I "];
wnam = [" BOB HUGO EINAR SABRINA "];
```

```
list tnam[3];
tnam[1] = wnam; tnam[2] = tnam[3] = rnam;
diss = dimname(diss,tnam);
print "Tensor diss=",diss;

 print "Example HV2 by Yoshio for Dissimilarity Data";
 print "Version 1: Use NLP with same parametrization as J. Ramsay";
 print "Iter=8 -118.146469  15.5943931  0.00302782";
 optn = [ "data"       "diss" ,
          "symm"              ,
          "vers"            1 ,
          "tech"     "trureg" ,
          "model"       "ide" ,
          "regres"    "power" ,
          "ndim"            2 ,
          "noplot"            ];
 < gof,conf,resi,wgt > = mds("mult",diss,optn);
 print "GOF=",gof;
 print "Configuration=",conf;
 print "Weights=",wgt;
 print "Residuals=",resi;

                      ******************************
                      3-Way Multidimensional Scaling
                   Method= MULTISCAL Algorithm Version 1
                      ******************************

                    Dissimilarity Input Data (Symmetric)
          Number Rows of Matrix . . . . . .          10
          Dimension . . . . . .           2
          Number Subjects / Replications. . . . . .           4
          Amount Input Information. . . . . .         180
          Number of Parameters. . . . . .          28
          Degrees of Freedom. . . . . .         152
          Maximum Number Iterations . . . . . .         200
Optimization Technique. . . . . .   TRUREG
FTOL Criterion for Main Iterations. . . . . .      0.0001
GTOL Criterion for Main Iterations. . . . . .      0.0001
Individual Metric Matrix. . . . . .   IDENTITY
Estimated Data Transformation Type. . . . . .     POWER
Stimulus Variance Type. . . . . .   CONSTANT
Subject Variance Type . . . . . .    SUBWISE
Assumed Error Distribution. . . . . .  LogNormal
Standard Error Weight Estimates . . . . . .   Post-Hoc
```

Initial Configuration Matrix

	D_1	D_2
A	0.859343210	-1.045559252
AA	2.230677017	0.440241710
E	1.416889423	-2.717584452
EE	3.446243955	0.930539614
O	-2.770119602	-0.116363392
OO	0.114727150	2.423379168
U	-2.590011936	0.945737375
UU	-0.759461217	3.485628578
MI	-2.430271423	-2.455526696
I	0.481983423	-1.890492653

****
Post-Mortem Display of Iterations
****

Iter	LogLikel.	ErrorSSQ	Grad_Length
1	-121.487981	18.4230946	8.772e-004
2	-119.798529	17.3913458	0.00188872
3	-119.034700	16.7870568	0.00137318
4	-118.592430	16.3915634	0.00103987
5	-118.357069	16.1125727	5.137e-004
6	-118.237378	15.9015657	3.484e-004
7	-118.177082	15.7324107	1.511e-004
8	-118.145825	15.5914118	1.282e-004
9	-118.128530	15.4710934	1.437e-004
...	...	...	...
21	-118.094170	14.8202012	1.443e-004
22	-118.093910	14.8007563	1.245e-004
23	-118.093714	14.7839044	1.074e-004
24	-118.093567	14.7693051	9.274e-005
25	-118.093456	14.7566612	8.008e-005
26	-118.093373	14.7457134	6.918e-005

****
ML Results for 2 Dimensions
****

Log Likelihood.	-118.093
AIC Statistic	290.187
BIC Statistic	376.397
Unbiased Standard Error Estimate.	0.439038
Number of Parameters.	27

```
Number of Degrees of Freedom for Error. . . . . .          153
Maximum Likelihood Standard Error Estimate. . . . . .    0.404773

                    ****************************
                    ML Results for Zero Dimensions
                    ****************************
Log Likelihood. . . . . .                               -202.607
AIC Statistic . . . . . .                                413.214
BIC Statistic . . . . . .                                425.986
Unbiased Standard Error Estimate. . . . . .              0.567709
Number of Parameters. . . . . .                                4
Number of Degrees of Freedom for Error. . . . . .            176

Chi Square for Comparing two Fits . . . . . .            169.027
Number of Degrees of Freedom for Chi Square . . . . . .       23
Probability of Exceeding this Chi Square. . . . . .            0

                 Rotated Final Configuration Matrix

                          D_1          D_2
                A     10.59324599 -5.216952244
                AA    15.83130929  18.26584612
                E     29.44523539 -4.618996044
                EE    35.47829438  9.266609904
                O    -34.00967817 -22.13379732
                OO   -2.923835976  39.40249724
                U    -35.82558566 -1.058964345
                UU   -33.85158441  27.65584174
                MI   -2.209476288 -46.21854398
                I     17.47207546 -15.34354108
          The decimal point has been moved 1 places to the left.

           ************************************************
           Standard Deviations and Variances for each Dimension
           ************************************************

                  Dim    Std.Dev.    Variance   Percent
                    1 25.28506093 639.3343062        53
                    2 23.76566795 564.8069729        46

                ***************************************
                Post-Hoc Standard Error Weight Estimates
                ***************************************

                    Input Order                    Size Order
```

```
     1          A  0.54516563     5          O  0.02632756
     2         AA  1.08746121     7          U  0.15617169
     3          E  0.21034545     3          E  0.21034545
     4         EE  1.87476471     1          A  0.54516563
     5          O  0.02632756     2         AA  1.08746121
     6         OO  1.48400630     8         UU  1.15398544
     7          U  0.15617169     6         OO  1.48400630
     8         UU  1.15398544    10          I  1.60610965
     9         MI  1.85566237     9         MI  1.85566237
    10          I  1.60610965     4         EE  1.87476471

        ************************************************
        Final Regression Coefficient and Exponent Estimates
        ************************************************

  Subject Regression   Exponent Reciprocal     Subject   Exponent
          Coefficient                                    by Size
1     BOB 0.69310405 1.37551454 0.72700068   2    HUGO 0.98393330
2    HUGO 1.56295494 0.98393330 1.01632905   3   EINAR 1.05422304
3   EINAR 1.04641567 1.05422304 0.94856587   4 SABRINA 1.27911146
4 SABRINA 0.88216714 1.27911146 0.78179270   1     BOB 1.37551454
          Mean= 1.1732 Standard Deviation= 0.184505

      ************************************************************
      Within-Subj. Std.Error Estimates and Multiple Correlations
      ************************************************************

        Standard Error                Multiple Correlation
Subj  Unordered Subj    Ordered   Subj  Unordered Subj    Ordered
   1       0.34    1       0.34      1       0.82    2       0.68
   2       0.52    3       0.35      2       0.68    4       0.78
   3       0.35    4       0.39      3       0.81    3       0.81
   4       0.39    2       0.52      4       0.78    1       0.82

print "Version 0: Optimization as with J. Ramsay original code";
print "5 -118.107126  14.4581275  0.00847371";
optn = [ "data"       "diss" ,
         "symm"              ,
         "model"       "ide" ,
         "regres"    "power" ,
         "ndim"            2 ,
         "pdis"              ,
         "ptab"              ,
         "pres"              ,
         "pcor"              ,
```

```
         "phis"            1 ];
< gof,conf,resi,wgt > = mds("mult",diss,optn);
print "GOF=",gof;
print "Configuration=",conf;
print "Weights=",wgt;
print "Residuals=",resi;
```

```
                    *******************************
                    Post-Mortem Display of Iterations
                    *******************************

                  Iter   LogLikel.    ErrorSSQ Grad_Length

                    0 -150.648557  21.2882303  2.22874509
                    1 -119.014014  14.7004026  0.01012332
                    2 -118.287445  14.6033957  0.00709952
                    3 -118.156586  14.6091758  0.01182026
                    4 -118.117461  14.6277275  0.00394852
                    5 -118.103169  14.6455720  0.00277625
                    6 -118.097349  14.6586806  0.00120595
                    7 -118.094928  14.6678452  8.934e-004
                    8 -118.093906  14.6736833  5.139e-004
                    9 -118.093466  14.6772521  3.501e-004
                   10 -118.093276  14.6792518  2.322e-004
                   11 -118.093192  14.6802316  1.520e-004

                    Rotated Final Configuration Matrix

                                  D_1          D_2
                      A     10.56743518 -5.200084827
                      AA    15.76972725  18.25724911
                      E     29.39649331 -4.600394462
                      EE    35.39606171  9.317637908
                      O    -33.91758115 -22.13270018
                      OO   -2.955787357  39.33476282
                      U    -35.76057138 -1.082939038
                      UU   -33.80350191  27.56071145
                      MI   -2.130204001 -46.12610694
                      I     17.43792835 -15.32813585
                  The decimal point has been moved 1 places to the right.
```

- Test Example: Unidimensional scaling

  The following is an example from text mining and is part of the `smacof` distribution in CRAN. It contains the relative frequencies of words (rows) used in 7 manuscripts (columns) of the ancient writer

Plato: "Republic", "Laws", "Critias", "Philebus", "Politicus", "Sophist", and "Timaeus". We are looking for a one dimensional scale of the manuscripts which hopefully correspoonds to the dates of its writings.

```
print "*** There are: 1*2*3...*7=5040 Permutations ***";
print "Nperm=5040 Nmon=5016 Nimpr=15 BestStress=0.257538";
rect = [   1  1.1 2.4 3.3 2.5 1.7 2.8 2.4 ,
           2  1.6 3.8 2.0 2.8 2.5 3.6 3.9 ,
           3  1.7 1.9 2.0 2.1 3.1 3.4 6.0 ,
           4  1.9 2.6 1.3 2.6 2.6 2.6 1.8 ,
           5  2.1 3.0 6.7 4.0 3.3 2.4 3.4 ,
           6  2.0 3.8 4.0 4.8 2.9 2.5 3.5 ,
           7  2.1 2.7 3.3 4.3 3.3 3.3 3.4 ,
           8  2.2 1.8 2.0 1.5 2.3 4.0 3.4 ,
           9  2.8 0.6 1.3 0.7 0.4 2.1 1.7 ,
          10  4.6 8.8 6.0 6.5 4.0 2.3 3.3 ,
          11  3.3 3.4 2.7 6.7 5.3 3.3 3.4 ,
          12  2.6 1.0 2.7 0.6 0.9 1.6 2.2 ,
          13  4.6 1.1 2.0 0.7 1.0 3.0 2.7 ,
          14  2.6 1.5 2.7 3.1 3.1 3.0 3.0 ,
          15  4.4 3.0 3.3 1.9 3.0 3.0 2.2 ,
          16  2.5 5.7 6.7 5.4 4.4 5.1 3.9 ,
          17  2.9 4.2 2.7 5.5 6.9 5.2 3.0 ,
          18  3.0 1.4 2.0 0.7 2.7 2.6 3.3 ,
          19  3.4 1.0 0.7 0.4 0.7 2.3 3.3 ,
          20  2.0 2.3 2.0 1.2 3.4 3.7 3.3 ,
          21  6.4 2.4 1.3 2.8 1.8 2.1 3.0 ,
          22  4.2 0.6 4.7 0.7 0.8 3.0 2.8 ,
          23  2.8 2.9 1.3 2.6 4.6 3.4 3.0 ,
          24  4.2 1.2 2.7 1.3 1.0 1.3 3.3 ,
          25  4.8 8.2 5.3 5.3 4.5 4.6 3.0 ,
          26  2.4 1.9 3.3 3.3 2.5 2.5 2.2 ,
          27  3.5 4.1 2.0 3.3 3.8 2.9 2.4 ,
          28  4.0 3.7 4.7 3.3 4.9 3.5 3.0 ,
          29  4.1 2.1 6.0 2.3 2.1 4.1 6.4 ,
          30  4.1 8.8 2.0 9.0 6.8 4.7 3.8 ,
          31  2.0 3.0 3.3 2.9 2.9 2.6 2.2 ,
          32  4.2 5.2 4.0 4.9 7.3 3.4 1.8 ];


  cnam = [ "Nobs" "Republic" "Laws" "Critias"
           "Philebus" "Politicus" "Sophist" "Timaeus" ];
  rect = cname(rect,cnam);
  rect = rect[,2:8 ]‘; print "Rectangular Data=", rect;
  data = dist(rect,"l2"); cnam = cnam[2:8];
  data = cname(data,cnam); data = rname(data,cnam);
  print "Data=", data;
```

```
print "Uniscale MDS with distance data";
optn = [ "data"       "dist" ,
         "print"           4 ,
         "pres"              ,
         "ptab"              ,
         "phis"            1 ];
< gof,conf,resi > = mds("unis",data,optn);
print "GOF=",gof;
print "X Configuration=",conf;
print "Resi=", resi;
```

```
              ******************************
              Metric Multidimensional Scaling
                      Method= UNIScale
              ******************************

              Distance Input Data (Symmetric)
 Number Rows of Matrix . . . . . .          7
 Dimension . . . . . .                      1
 Amount Input Information. . . . . .       21
 Number of Parameters. . . . . .            7
 Degrees of Freedom. . . . . .             14
 Maximum Number Iterations . . . . . .    400

                Distance Matrix: Subject=1

              Republic         Laws      Critias     Philebus
Republic             0
Laws        12.27843638            0
Critias     10.72054103  11.80296573            0
Philebus    12.64990119  6.437390776  11.75372281            0
Politicus   11.54816003  8.584870413  11.96285919  6.991423317
Sophist     8.589528509  10.58300524  9.377099765  10.08265838
Timaeus     8.889881889  13.01960061  9.517352573  12.26254460

                Distance Matrix: Subject=1

                  Politicus      Sophist      Timaeus
      Politicus            0
      Sophist     7.722693831            0
      Timaeus     11.23076133  6.098360435            0
                  Additive Constant: 0
```

History of Iteration
****

N_Iter	Improve	Stress
1	1	0.494971819
18	18	0.492492433
24	24	0.488592955
49	49	0.465981841
73	73	0.450675119
97	97	0.439404814
264	264	0.435612187
360	360	0.435357966
377	377	0.403376562
378	378	0.388217810
384	384	0.383258851
503	502	0.382353815
504	503	0.366326162
526	524	10 Percent
624	621	0.360591889
1222	1217	20 Percent
1344	1337	0.358843869
1942	1933	30 Percent
2955	2942	40 Percent
3675	3658	50 Percent
4395	4374	60 Percent

Nperm=5040 Nmon=5016 Nimpr=15 BestStress=0.358844

Final X Configuration Matrix

	Republic	Laws	Critias	Philebus	Politicus
1	-0.691868608	1.003449310	-1.042307431	0.756958154	0.430274782

Final X Configuration Matrix

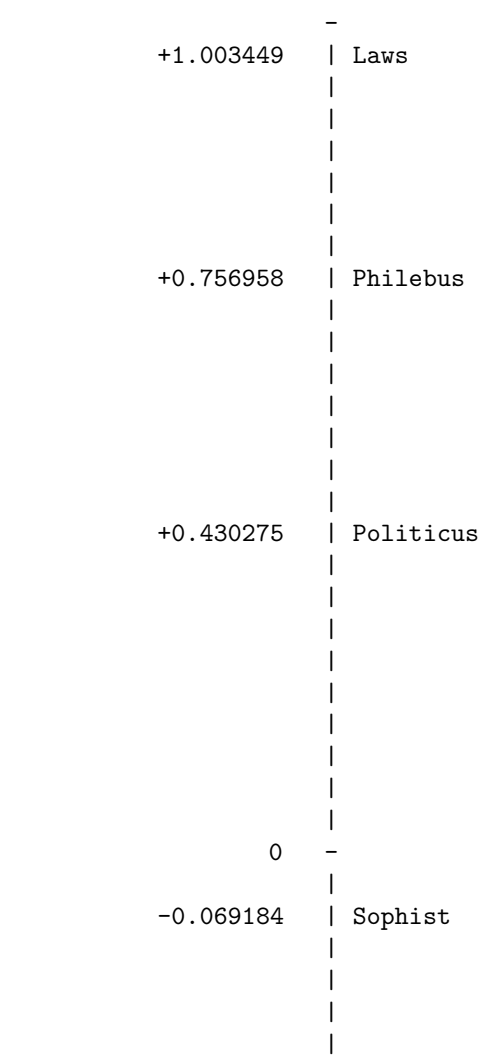
	Sophist	Timaeus
1	-0.069184173	-0.387322034

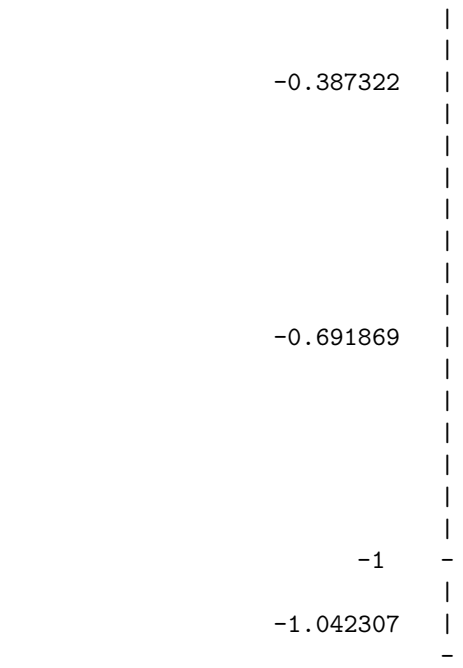
"Sophist" separates two groups of three manuscripts with positive and negative coordinates.

Sorted Configuration X
****

3	Critias	-1.042
1	Republic	-0.692
7	Timaeus	-0.387
6	Sophist	-0.069
5	Politicus	0.430
4	Philebus	0.757
2	Laws	1.003

Plot of Onedimensional Configuration
****





Model Distance Matrix

	Republic	Laws	Critias	Philebus
Republic	0			
Laws	1.695317917	0		
Critias	0.350438823	2.045756741	0	
Philebus	1.448826761	0.246491156	1.799265585	0
Politicus	1.122143390	0.573174527	1.472582213	0.326683372
Sophist	0.622684434	1.072633483	0.973123258	0.826142327
Timaeus	0.304546574	1.390771344	0.654985397	1.144280188

Model Distance Matrix

	Politicus	Sophist	Timaeus
Politicus	0		
Sophist	0.499458955	0	
Timaeus	0.817596816	0.318137861	0

Residual Matrix

	Republic	Laws	Critias	Philebus
Republic	0			

```
Laws       -0.319928810           0
Critias     0.850440092 -0.723628255           0
Phileb...  -0.031827417  0.474603687 -0.482653124           0
Politicus   0.171442619  0.388473755 -0.132543000  0.456472332
Sophist     0.339485632  0.112839137  0.077267890  0.303283121
Timaeus     0.691268067  0.067640486  0.411116398  0.229328776
```

Residual Matrix

```
                Politicus      Sophist      Timaeus
Politicus               0
Sophist     0.365611209            0
Timaeus     0.440435259  0.364979945            0
```

The 1-dimensional TORGERSON result is very different of that of the UNISCALE method:

Final X Configuration Matrix

```
                    D_1
Republic     7.520698269
Laws         4.972851713
Critias      7.742108379
Phileb...    0.296523444
Politicus   -5.346583361
Sophist     -16.72126524
Timaeus      1.535666794
```

However, the 1-dimensional ALSCAL result is rather similar of that of the UNISCALE method:

Final X Configuration Matrix

```
                    D_1
Republic     6.043733493
Laws        -6.974437017
Critias      5.605827640
Phileb...   -6.708128627
Politicus   -5.598557811
Sophist      1.772787456
Timaeus      5.858774866
```

We can see, that "Sophist" separates two groups of three manuscripts with either positive or negative coordinates. That is also so for the results of KYST and MULTISCALE.

The 1-dimensional KYST result is:

Estimated X Configuration Matrix

	D_1
Republic	-1.137760957
Laws	1.356688108
Critias	-0.620375402
Philebus	1.137828014
Politicus	0.701290151
Sophist	-0.144324779
Timaeus	-1.293345135

The 1-dimensional MULTISCALE result is:

Unrotated Common Space Configuration

	D_1
Republic	-8.725761724
Laws	4.792481142
Critias	-4.193938011
Philebus	12.89588483
Politicus	8.729151398
Sophist	-0.001702011
Timaeus	-13.45554400

5 New Developments

5.1 The `mdu()` Function

< gof,xconf,yconf,resi,... > = mdu("meth",data,optn<,xini,<,yini,..> . >)

< gof,xconf,yconf,resi,... > = mdu("torg",data,optn<,...> . >) TORGERSON

< gof,xconf,yconf,resi,... > = mdu("alsc",data,optn<,...> . >) ALSCAL

< gof,xconf,yconf,resi,... > = mdu("smac",data,optn<,...> . >) SMACOF

Purpose: This function implements a variety of algorithms for computing multidimensional unfolding. The input `data` matrix **A** is usually $m \times n$ rectangular and estimated are two point configurations, the $m \times d$ configuration $X = (x_{ik})$ for the m rows and the $n \times d$ configuration $Y = (y_{jk})$ for the n columns of the data. You are recommended that $m \geq n$, i.e. input matrix **A** has more rows than columns. Otherwise **A** is transposed (indicated by a message in the log output), and the output maybe confusing.

The following estimation methods are implemented:

TORG Torgerson metric unfolding which is commonly used for obtaining starting configurations for other, more sophisticated methods of `mds()` and `mdu()`.

ALSCAL Takane, Young, & de Leeuw (1977) metric unfolding (see de Leeuw, 2012)

SMACOF metric and nonmetric method by de Leeuw (1984, 1994, 2007): which is an iterative ALS methods suitable for problems with many observations

Input: data This function permits the following forms of data input:

1. $m \times n$ matrices of *raw* data.
2. $m \times n$ matrices of distance (dissimilarity) data.

optn This argument must be specified in form of a two column matrix where the first column defines the option as string value (in quotes) and the second column can be used for a numeric or string specification of the option. See table below for content.

xini an $n \times d$ matrix of initial values for the **X** configuration of points

Option	Second Column	Meaning
"addc"		add constant if distance data do not satisfy triangle inequality
"dimmax"	int	maximum number dimensions d (def=2)
"dimmin"	int	minimum number dimensions d (def=2)
"dimdif"	int	step size dimensions d (def=1)
"data"	string	kind of input data
	"raw"	$n \times p$ raw data
	"diss"	dissimilarities
	"simi"	similarities
	"dist"	distances
	"corr"	correlations
	"cova"	covariances
	"scp"	scalar product data
"gutlin"		apply Guttman-Lingoes transformation
"met"	string	estimation method
	"torg"	Torgerson
	"alsc"	ALSCAL
	"smac"	SMACOF

Option	Second Column	Meaning
"maxit"	int	number of iterations
"ndim"	int	number dimensions d (def=2)
"noplot"		do not plot
"nopr"		do not print
"pdat"		print input data
"pdis"		print distance matrices
"pcoef"		print X and W configurations
"phis"		print iteration history
"print"	int	controls printed output
"pres"		printed observationwise output
"plotf"	char	plotting into text file (with larger line size)
"plotco"	char	configuration plotting (dimensions pairwise)
	"non"	
	"som"	
	"all"	
"plotsc"	char	scatter plotting (data vs. model)
	"non"	
	"som"	
	"all"	
	"joi"	
"plowgt"	char	subject weights plotting (dimensions pairwise)
	"non"	
	"som"	
	"all"	
"seed"	int	seed of random generator (def: time of day)
"sing"	real	singularity criterion (def=1.e-8)
"xtol"	real	termination criterion (def=1.e-5)

Specific options for the ALSCAL method:

Option	Second Column	Meaning
`"ftol"`	real	termination test for stress value, def=.0001

Specific options for the SMACOF method:

Option	Second Column	Meaning

Output: **gof** vector of scalar results

xconf the $m \times d$ configuration matrix $\mathbf{X}$

yconf the $n \times d$ configuration matrix $\mathbf{Y}$

Restrictions: 1. Since there are more, i.e. $(m+n)*d$, parameters (X, Y) to estimate from only $m \times n$ data, the dimension d of the two point configurations cannot exceed the upper limit

$$d < \frac{m*n}{m+n}$$

2.

Relationships: mds(), dist(), cluster(), anacor()

Examples: 1. TORGERSON with dissimilarity data:

```
srand(123);

/* Normal random generated nxnd configuration: X[nr,nc] */
nr = 10; nc = 4;
coef  = rand(nr,nc,"g","norm",0.,1.);
coef += rand(nr,nc,"g","norm",1.,1.);
mu = cons(nc,1,1.); id = ide(nc);
for (j = 1; j <= nr; j++) coef[j,] += mrand("mnor",mu,id)';
                                    print "Normal: Unscaled coef=",coef;
/*--- CMAT code for unsymmetric ALSCAL: data must be nonnegative ---*/
data = coef .* coef; wgt = cons(nr,nc,1.);
```

```
Normal: Unscaled coef=
    |         1         2         3         4
----------------------------------------------
  1 |   0.13945    5.6162    4.9593   0.89268
  2 |   0.46138    3.9600    2.1125  -0.30472
  3 |   0.24811    3.6825  -0.03017    3.1384
  4 |    3.0081   0.83865    3.5334  -0.26173
  5 |   0.38527    1.5628    2.9947    3.1497
  6 |    1.3847    2.1498   0.19754    4.5737
  7 |    3.0734    3.7302    2.1824    3.4404
  8 |    2.0061    3.5394    3.2046    2.3395
  9 |   0.69489    1.5902    1.5327   -1.9894
 10 |    2.1309    1.0724    3.1897    2.5358
```

```
MDU Input Data=
    |         1         2         3         4
----------------------------------------------
  1 |    0.0194    31.542    24.595   0.79688
  2 |   0.21288    15.681    4.4628    0.0929
  3 |    0.0616    13.561    0.0009    9.8495
  4 |    9.0488   0.70334    12.485    0.0685
  5 |   0.14843    2.4422    8.9681    9.9209
  6 |    1.9174    4.6217    0.0390    20.919
  7 |    9.4458    13.915    4.7630    11.836
  8 |    4.0243    12.527    10.269    5.4732
  9 |   0.48287    2.5288    2.3493    3.9578
```

```
print "Torgerson MDU with dissimilarity data";
optn = [ "data"       "diss" ,
         "dimmax"          2 ,
         "noplot"            ,
         "phis"             ];
< gof,xcof,ycof,resi > = mdu("torg",data,optn);
print "GOF=",gof;
print "X Configuration=", xcof;
print "Y Configuration=", ycof;
print "Residuals=", resi;
```

```
                ********************************
                 Metric Multidimensional Unfolding
                 ********************************

              Dissimilarity Input Data (Unsymmetric)
   Number Rows of Matrix . . . . . .          10
   Number Columns of Matrix  . . . . .         4
   Dimension . . . . . .                       2
   Amount Input Information. . . . . .        45
   Number of Parameters. . . . . .            20
   Degrees of Freedom. . . . . .              25

                     Dissimilarity Matrix: 1

                  V_1          V_2          V_3          V_4
    O_01   0.019445908  31.54219663  24.59487246  0.796877292
    O_02   0.212875899  15.68138411  4.462843391  0.092851552
    O_03   0.061558079  13.56062791  0.000910512  9.849452952
    O_04   9.048774716  0.703336730  12.48465067  0.068502430
    O_05   0.148432244  2.442195236  8.968070788  9.920877450
```

```
O_06   1.917363674  4.621667831  0.039023687  20.91914540
O_07   9.445831916  13.91458893  4.763041460  11.83606652
O_08   4.024251501  12.52734359  10.26943488  5.473213815
O_09   0.482873979  2.528836104  2.349277526  3.957779069
O_10   4.540693885  1.150088809  10.17405088  6.430341459
        Additive Constant: 31.5422


      ******************************
       Compute Metric MDU by Torgerson
       ******************************

         Final X Configuration Matrix

                    DIM_1        DIM_2
       O_01   0.614786087 -0.627801702
       O_02   0.187224292 -0.187592207
       O_03   0.226072154  0.154744564
       O_04   0.230369607  0.045581125
       O_05   0.208774640  0.233656348
       O_06   0.283866494  0.609921152
       O_07   0.436578890  0.246394216
       O_08   0.337016198 -0.016910477
       O_09   0.062635828  0.144452288
       O_10   0.224799973  0.190234144


         Final Y Configuration Matrix

                    DIM_1        DIM_2
        V_1   21.36554454 -4.470173760
        V_2   89.04503943  6.115758264
        V_3   69.19377362  4.115629338
        V_4   63.34790991 -11.58437848
```

2. ALSCAL with dissimilarity data: Using the same data as with Torgerson method we run the metric rectangular ALSCAL algotrithm with `mdu()`:

```
print "ALSCAL MDU with dissimilarity data";
optn = [ "data"       "diss" ,
         "dimmax"          2 ,
         "noplot"            ,
         "phis"              ];
< gof,xcof,ycof,resi > = mdu("alsc",data,optn);
print "GOF=",gof;
```

```
print "X Configuration=", xcof;
print "Y Configuration=", ycof;
print "Residuals=", resi;
```

```
                ********************************
                Metric Multidimensional Unfolding
                ********************************

           Dissimilarity Input Data (Unsymmetric)
 Number Rows of Matrix . . . . . .          10
 Number Columns of Matrix  . . . . .         4
 Dimension . . . . . .                       2
 Amount Input Information. . . . . .        45
 Number of Parameters. . . . . .            20
 Degrees of Freedom. . . . . .              25
 Maximum Number Iterations . . . . . .    1000

                  Dissimilarity Matrix: 1

               V_1          V_2          V_3          V_4
 O_01   0.019445908  31.54219663  24.59487246  0.796877292
 O_02   0.212875899  15.68138411  4.462843391  0.092851552
 O_03   0.061558079  13.56062791  0.000910512  9.849452952
 O_04   9.048774716  0.703336730  12.48465067  0.068502430
 O_05   0.148432244  2.442195236  8.968070788  9.920877450
 O_06   1.917363674  4.621667831  0.039023687  20.91914540
 O_07   9.445831916  13.91458893  4.763041460  11.83606652
 O_08   4.024251501  12.52734359  10.26943488  5.473213815
 O_09   0.482873979  2.528836104  2.349277526  3.957779069
 O_10   4.540693885  1.150088809  10.17405088  6.430341459
                Additive Constant: 31.5422

  ******************************************************
  Compute ALSCAL MDU by Takane, Young, and de Leeuw, 1977
  ******************************************************

            Iteration       Stress   DiffStress

                    1  877.0948534  511071997.0
                    2  325.3357999  551.7590535
                    3  265.4203366  59.91546331
                    4  193.4231729  71.99716371
                    5  158.6102045  34.81296839
                    6  143.4233042  15.18690030
```

7	138.4117298	5.011574408
8	137.0035431	1.408186701
9	136.5782488	0.425294284
10	136.3962784	0.181970452
11	136.2951587	0.101119612
12	136.2337215	0.061437210
13	136.1957329	0.037988638
14	136.1722208	0.023512143
15	136.1576740	0.014546799
16	136.1486635	0.009010502
17	136.1430672	0.005596283
18	136.1395784	0.003488725
19	136.1373942	0.002184234
20	136.1360205	0.001373721
21	136.1351526	0.000867887
22	136.1346019	0.000550710
23	136.1342510	0.000350884
24	136.1340266	0.000224412
25	136.1338826	0.000144019
26	136.1337899	9.2710e-005

Final Configuration Matrix

	DIM_1	DIM_2
O_01	55.02690109	-2.756160458
O_02	55.97579511	-0.811575552
O_03	57.02836252	0.195408961
O_04	53.62051950	0.666187285
O_05	54.32574967	1.147787135
O_06	56.72928104	2.000791735
O_07	57.68775782	0.390559253
O_08	54.09925512	-0.378547704
O_09	55.24357386	0.724626746
O_10	53.95017952	1.065336403

Final Configuration Matrix

	DIM_1	DIM_2
V_1	55.62672827	-0.409684457
V_2	54.52200152	2.729677275
V_3	56.97073847	1.594754255
V_4	54.52073215	-1.567429568

Residuals=

```
          |      DA_21      FD_21      DI_21      RS_21
----------------------------------------------------------
    O_0101 |     0.0194     17.828     5.8657    -5.8463
    O_0102 |     31.542     51.025     30.349     1.1929
    O_0103 |     24.595     39.857     22.709     1.8859
    O_0104 |    0.79688     45.519     1.6693   -0.87241
    O_0201 |    0.21288     4.0869    0.28336    -0.0705
    O_0202 |     15.681     13.932     14.654     1.0274
..........................................................
    O_0903 |     2.3493     4.4732     3.7402    -1.3909
    O_0904 |     3.9578     2.6484     5.7760    -1.8182
    O_1001 |     4.5407     2.6127     4.9865   -0.44581
    O_1002 |     1.1501     23.474     3.0970    -1.9469
    O_1003 |     10.174     13.230     9.4041    0.76999
    O_1004 |     6.4303     12.660     7.2570   -0.82665
```

```
di = cons(nr,nc,.);
for (i = 1; i <= nr; i++)
for (j = 1; j <= nc; j++) {
  d1 = xcof[i,1] - ycof[j,1];
  d2 = xcof[i,2] - ycof[j,2];
  di[i,j] = d1 * d1 + d2 * d2;
}
print "Squared Euclidean Distances", di;
```

```
Squared Euclidean Distances
    |         1         2         3         4
-------------------------------------------------
  1 |    5.8657    30.349    22.709    1.6693
  2 |   0.28336    14.654    6.7803    2.6885
  3 |    2.3307    12.704    1.9615    9.3958
  4 |    5.1824    5.0707    12.086    5.7994
  5 |    4.1183    2.5409    7.1957    7.4104
  6 |    7.0260    5.4034   0.22317    17.610
  7 |    4.8882    15.493    1.9642    13.864
  8 |    2.3341    9.8398    12.139    1.5911
  9 |    1.4335    4.5409    3.7402    5.7760
 10 |    4.9865    3.0970    9.4041    7.2570
```

5.2 The `xplot()` Function

xplot($x <, titl <, optn <, labl <, fpath >>>>$)

Purpose: The `xylot` function prints one or more onedimensional x plots for specified vectors x.

Input: **x** This argument must be a vector or a matrix with c_x columns.

titl This argument must be a single string argument specifying the title for the plot.

optn This argument must be a numeric scalar or a vector specifying the output options:

1 amount of printed output (=0: no printed output)

2 =0 for vertical plot on one page (default), =1: for horicontal plot.

3 specifying how to place the labels if they are specified:

4

fpath This argument must be a single string argument specifying the (path and) file name for a text output file containing the plot.

Output: This function has no return arguments. However, it has printed and/or file output.

Restrictions: 1. Missing values in x are not permitted.

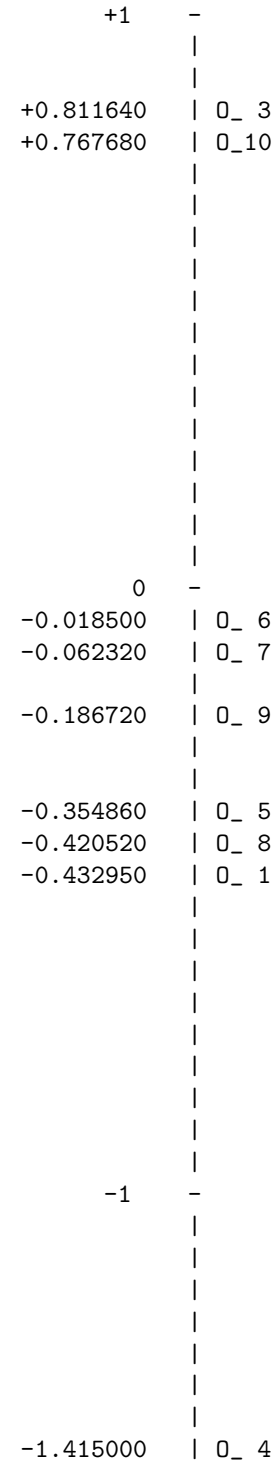
2. String or complex data for x are not permitted.

Relationships: xyplot(), histplot() gnuplot

Examples: 1. Vertical Plot:

```
data = [ -0.43295  1.3116   0.81164  -1.4150  -0.35486
         -0.01850 -0.06232 -0.42052  -0.18672  0.76768 ];
cnam = [" O_01 O_02 O_03 O_04 O_05 O_06 O_07 O_08 O_09 O_10 "];
data = cname(data,cnam);


optn = [ 2 ,
         0 ];
xplot(data,titl,optn);
```

```
                    [0] Vertical Univariate Plot
                    ****************************

                                        -
                       +1.311600        | O_ 2
                                        |
                                        |
                                        |
                                        |
                                        |
```


+1
+0.811640
+0.767680
0
-0.018500
-0.062320
-0.186720
-0.354860
-0.420520
-0.432950
-1
-1.415000

2. Multiply with larger scale:

```
dat1000 = 1000. * data;
optn = [ 2 ,
         0 ];
xplot(dat1000,titl,optn);
```

```
            [0] Vertical Univariate Plot: fact=1000
            ***************************************

                                  -
                       +1311.6000 | O_ 2
                                  |
                                  |
                                  |
                                  |
                                  |
                           +1000  -
                                  |
                                  |
                        +811.6400 | O_ 3
                        +767.6800 | O_10
                                  |
                                  |
                                  |
                                  |
                                  |
                                  |
                                  |
                                  |
                                  |
                                  |
                                  |
                                  |
                                  |
                                0 -
                         -18.5000 | O_ 6
                         -62.3200 | O_ 7
                                  |
                        -186.7200 | O_ 9
                                  |
                                  |
                        -354.8600 | O_ 5
```

```
                    -420.5200  | O_ 8
                    -432.9500  | O_ 1
                               |
                               |
                               |
                               |
                               |
                               |
                               |
                               |
                               |
                               |
                        -1000  -
                               |
                               |
                               |
                               |
                               |
                               |
                               |
                               |
                  -1415.0000   | O_ 4
                               -
```

3. Horicontal Plot:

```
titl = "[1] Horicontal Univariate Plot";
optn = [ 2 ,
         1 ];
xplot(data,titl,optn);
```

```
                  [1] Horicontal Univariate Plot
                  ******************************

-         -           -- -  -  --                ++    +      +
1         1           00 0  0  000               00    1      1
.                     .. .  .  ..                ..           .
4                     44 3  1  00                78           3
1                     32 5  8  61                61           1
5                     20 4  6  28                71           1


|x---------|-----------xx-x--x--xx0----------------xx----|------x|
 0                    00 0  0  00                00           0
 -                    -- -  -  --                --           -
                                                 1
 4                    18 5  9  76                03           2
```

4. Multiply with larger scale:

```
titl = "[1] Horicontal Univariate Plot";
optn = [ 2 ,
         1 ];
dat1000 = 1000. * data;
xplot(dat1000,titl,optn);
```

```
                  [1] Horicontal Univariate Plot
                  ******************************

-         -           -- -  -  --                ++    +      +
1         1           44 3  1  610               78    1      1
4         0           32 5  8  28                61    0      3
1         0           20 4  6  ..                71    0      1
5         0           .. .  .  35                ..    0      1
.                     95 8  7  20                66           .
0                     52 6  2  00                84           6

|x---------|----------xx-x--x--xx0---------------xx----|------x|
 0                    00 0  0  00                00           0
 -                    -- -  -  --                --           -
                                                 1
 4                    18 5  9  76                03           2
```

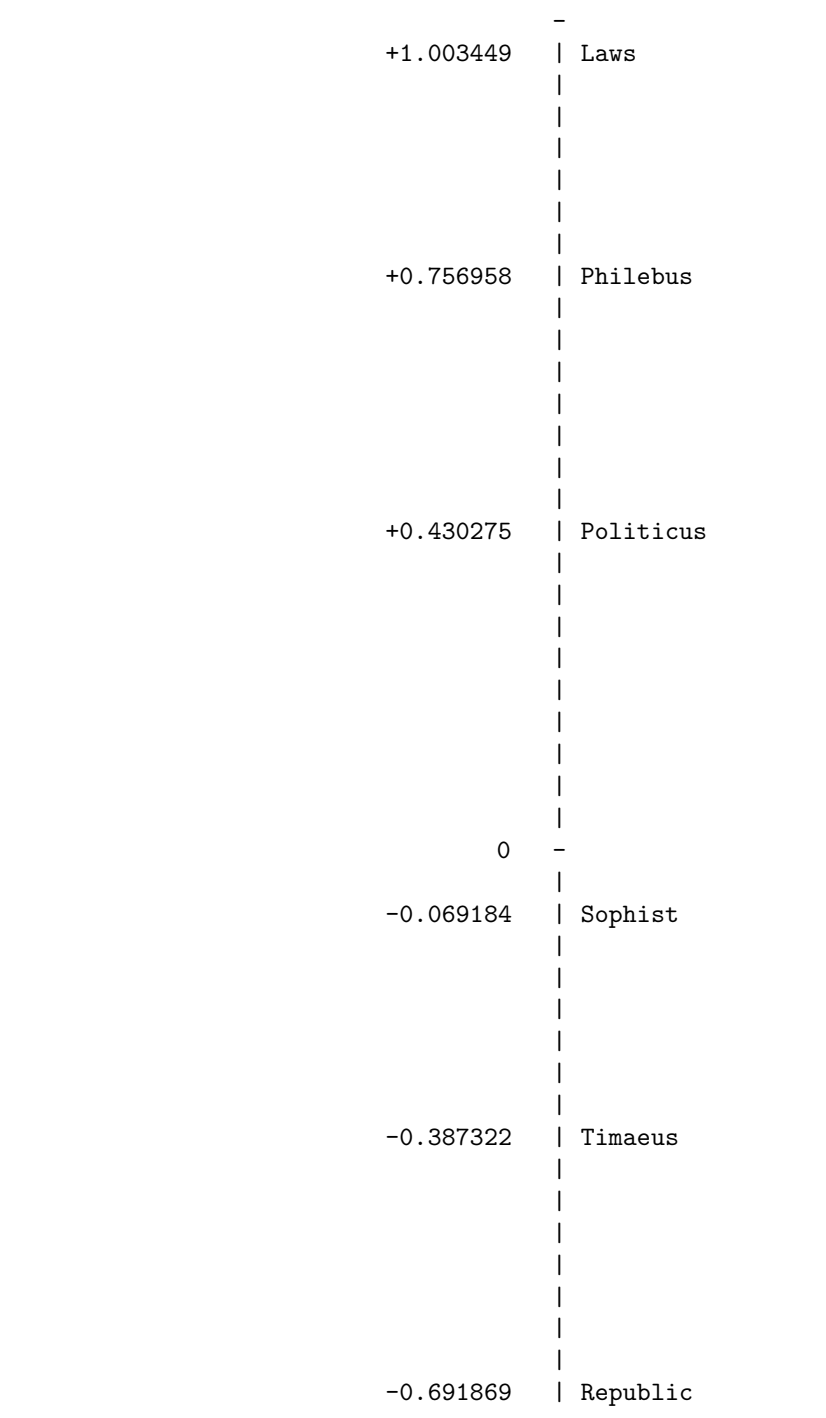
5. One dimensional scaling of works by Platon:

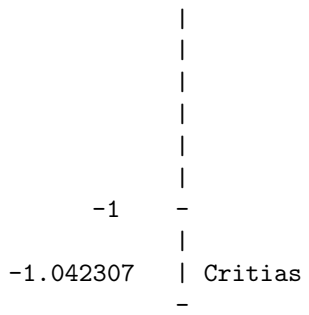
```
print "*** There are: 1*2*3...*7=5040 Permutations ***";
print "Nperm=5040 Nmon=5016 Nimpr=15 BestStress=0.257538";
rect = [   1  1.1 2.4 3.3 2.5 1.7 2.8 2.4 ,
           2  1.6 3.8 2.0 2.8 2.5 3.6 3.9 ,
           3  1.7 1.9 2.0 2.1 3.1 3.4 6.0 ,
           4  1.9 2.6 1.3 2.6 2.6 2.6 1.8 ,
           5  2.1 3.0 6.7 4.0 3.3 2.4 3.4 ,
           6  2.0 3.8 4.0 4.8 2.9 2.5 3.5 ,
           7  2.1 2.7 3.3 4.3 3.3 3.3 3.4 ,
           8  2.2 1.8 2.0 1.5 2.3 4.0 3.4 ,
           9  2.8 0.6 1.3 0.7 0.4 2.1 1.7 ,
          10  4.6 8.8 6.0 6.5 4.0 2.3 3.3 ,
          11  3.3 3.4 2.7 6.7 5.3 3.3 3.4 ,
          12  2.6 1.0 2.7 0.6 0.9 1.6 2.2 ,
          13  4.6 1.1 2.0 0.7 1.0 3.0 2.7 ,
          14  2.6 1.5 2.7 3.1 3.1 3.0 3.0 ,
```

```
         15  4.4 3.0 3.3 1.9 3.0 3.0 2.2 ,
         16  2.5 5.7 6.7 5.4 4.4 5.1 3.9 ,
         17  2.9 4.2 2.7 5.5 6.9 5.2 3.0 ,
         18  3.0 1.4 2.0 0.7 2.7 2.6 3.3 ,
         19  3.4 1.0 0.7 0.4 0.7 2.3 3.3 ,
         20  2.0 2.3 2.0 1.2 3.4 3.7 3.3 ,
         21  6.4 2.4 1.3 2.8 1.8 2.1 3.0 ,
         22  4.2 0.6 4.7 0.7 0.8 3.0 2.8 ,
         23  2.8 2.9 1.3 2.6 4.6 3.4 3.0 ,
         24  4.2 1.2 2.7 1.3 1.0 1.3 3.3 ,
         25  4.8 8.2 5.3 5.3 4.5 4.6 3.0 ,
         26  2.4 1.9 3.3 3.3 2.5 2.5 2.2 ,
         27  3.5 4.1 2.0 3.3 3.8 2.9 2.4 ,
         28  4.0 3.7 4.7 3.3 4.9 3.5 3.0 ,
         29  4.1 2.1 6.0 2.3 2.1 4.1 6.4 ,
         30  4.1 8.8 2.0 9.0 6.8 4.7 3.8 ,
         31  2.0 3.0 3.3 2.9 2.9 2.6 2.2 ,
         32  4.2 5.2 4.0 4.9 7.3 3.4 1.8 ];

 cnam = [ "Nobs" "Republic" "Laws" "Critias"
          "Philebus" "Politicus" "Sophist" "Timaeus" ];
 rect = cname(rect,cnam);
 rect = rect[,2:8 ]‘; print "Rectangular Data=", rect;
 data = dist(rect,"l2"); cnam = cnam[2:8];
 data = cname(data,cnam); data = rname(data,cnam);
 print "Data=", data;


 print "Uniscale MDS with distance data";
 optn = [ "data"       "dist" ,
          "print"           4 ,
          "pres"              ,
          "ptab"              ,
          "phis"            1 ];
 < gof,conf,resi > = mds("unis",data,optn);
 print "GOF=",gof;
 print "X Configuration=",conf;


titl = "Plot Univariate MDS vertically";
optn = [ 2 ,
         0 ];
xplot(conf,titl,optn);
```

Plot of Onedimensional Configuration

****


+1.003449
Laws
+0.756958
Philebus
+0.430275
Politicus
0
-0.069184
Sophist
-0.387322
Timaeus
-0.691869
Republic


-1
-1.042307
Critias

6 Illustrations

7 The Bibliography

References

[1] Borg, I., Groenen, P.J.F., & Mair, P. (2010), "Multidimensionale Skalierung", Muenchen: Hampp Verlag.

[2] de Leeuw, J. (2005), "SMACOF in R", Technical Report.

[3] de Leeuw, J. (2012), "ALSCAL in R", Technical Report.

[4] de Leeuw, J. (2017), "Pseudo confidence regions for MDS", Technical Report.

[5] de Leeuw, J. & Meulman, J. (1986), "A special Jackknife for multidimensional scaling", *Journal of Classification*, **3**, 97-112.

[6] Gifi, A., (1990) *Nonlinear Multivariate Analysis*, Chichester: Wiley, 1990.

[7] Jeffrey, D.J. (1997), "Formulae, algorithms, and quartic extrema"; *Mathematics Magazine*, **70**, pp. 349 - 356.

[8] Kruskal, J. B., Young, F. W. & Seery, J. B. (1978), "How to use KYST, a very flexible program to do multidimensional scaling and unfolding"; Technical Report, Murray Hill: Bell Laboratories.

[9] Mair, P., de Leeuw, J., Borg, I., & Groenen, P.J.F. (2016), "Multidimnensional scaling in R: SMACOF", Softwarepackage available in CRAN.

[10] Ramsay, J.O. (1977), "Maximum likelihood estimation in multidimensional scaling"; *Psychometrika*, **42**, 241-266.

[11] Ramsay, J.O. (1978), "Confidence regions for multidimensional scaling analysis"; *Psychometrika*, **43**, 145-160.

[12] Ramsay, J.O. (1980), "Some small sample results for maximum likelihood estimation in multidimensional scaling"; *Psychometrika*, **45**, 139-144.

[13] Ramsay, J.O. (1982), "Some statistical approaches to multidimensional scaling"; *Journal of the Royal Statistical Society, Series A*, **145**, 285-312.

[14] Takane, Y. (1977), "On the relations among four methods of multidimensional scaling"; *Behaviometrika*, **4**,29-42.

[15] Takane, Y., Young, F. W. & de Leeuw, J. (1977), "Nonmetric individual differences multidimensional scaling: An alternating least squares method with optimal scaling features"; *Psychometrika*, **42**, 7-67.

[16] Young, F. W., Takane, Y. & Lewyckyj, R. (1978), "ALSCAL: A nonmetric multidimensional program with several individual-differences options",