



THE  
POWER  
TO KNOW.

**SAS<sup>®</sup> Enterprise Miner<sup>™</sup> and  
SAS<sup>®</sup> Text Miner Procedures  
Reference for SAS<sup>®</sup> 9.1.3  
The SPSVD Procedure  
(Book Excerpt)**

The correct bibliographic citation for this manual is as follows: SAS Institute Inc., SAS® *Enterprise Miner™* and SAS® *Text Miner Procedures: Reference for SAS® 9.1.3*, Cary, NC: SAS Institute Inc.

**SAS® Enterprise Miner™ and SAS® Text Miner Procedures: Reference for SAS 9.1.3**

Copyright © 2008 by SAS Institute Inc., Cary, NC, USA.

All rights reserved. Produced in the United States of America.

**For a Web download or e-book:** Your use of this publication shall be governed by the terms established by the vendor at the time you acquire this publication.

**U.S. Government Restricted Rights Notice.** Use, duplication, or disclosure of this software and related documentation by the U.S. government is subject to the Agreement with SAS Institute and the restrictions set forth in FAR 52.227-19 Commercial Computer Software-Restricted Rights (June 1987).

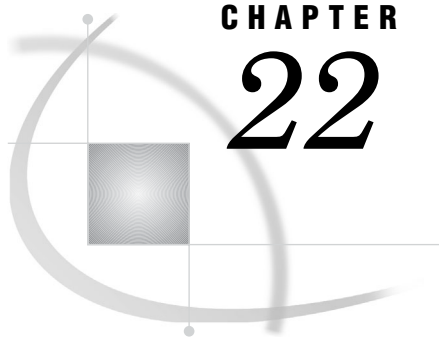
SAS Institute Inc., SAS Campus Drive, Cary, North Carolina 27513.

1st electronic book, October 2008

SAS® Publishing provides a complete selection of books and electronic products to help customers use SAS software to its fullest potential. For more information about our e-books, e-learning products, CDs, and hard-copy books, visit the SAS Publishing Web site at [support.sas.com/publishing](http://support.sas.com/publishing) or call 1-800-727-3228.

SAS® and all other SAS Institute Inc. product or service names are registered trademarks or trademarks of SAS Institute Inc. in the USA and other countries. ® indicates USA registration.

Other brand and product names are registered trademarks or trademarks of their respective companies.



## CHAPTER

## 22

## The SPSVD Procedure

---

<i>Overview: SPSVD Procedure</i>	441
<i>Syntax: SPSVD Procedure</i>	443
<i>PROC SPSVD Statement</i>	443
<i>COL Statement</i>	446
<i>ENTRY Statement</i>	446
<i>OUTPUT Statement</i>	446
<i>ROW Statement</i>	448
<i>Example: SPSVD Procedure</i>	448
<i>Example 1: PROC SPSVD</i>	448
<i>References</i>	449

---

### Overview: SPSVD Procedure

The SPSVD procedure has two main functions. The first is to calculate a truncated singular value decomposition of a large sparse matrix. The second is to project the rows or columns of a sparse matrix onto the columns of a dense matrix.

The SPSVD procedure takes at least one data set as input. This data set must be a compressed representation of a sparse matrix. The data set must contain at least three variables, with one observation for each non-zero entry in the matrix. One variable indicates the row number, another variable indicates the column number and a third indicates the entry. Thus the following matrix

$$\begin{bmatrix} 1 & 0 & 0 & 2 \\ 0 & 3 & 0 & 0 \\ 2 & 0 & 1 & 1 \end{bmatrix}$$

would be represented in compressed form as:

<i>Row</i>	<i>Column</i>	<i>Entry</i>
1	1	1
1	4	2
2	2	3
3	1	2
3	3	1
3	4	1

The last observation in this data set indicates that the number 1 appears in the 3rd row and 4th column of the original matrix. The Text Parsing node in Enterprise Miner

produces a data set that is in the correct format. For text mining we generally consider the terms as rows and the documents as columns. The number of times that the term appears in the document is the entry. Thus if term 5 appears in document 10 three times we would say there is a 3 in row 5, column 10. These correspond to the numeric key for the term, the document number, and the number of times that the term appears in the current document. Note that other matrices can be passed to the procedure (via the IN\_U option for example) and these matrices are not passed in the sparse matrix format.

The SPSVD procedure computes a truncated singular value decomposition of the sparse matrix. You can specify the value for  $k$  with a procedure option (MAX\_k) or have the procedure recommend the number of dimensions to use. In either case, if we call the sparse matrix  $A$ , the SPSVD procedure will compute the leading columns of three new matrices,  $U$ ,  $\Sigma$ , and  $V$  such that  $A = U\Sigma V^T$ . Since only the first  $k$  (or MAX\_k) columns of these matrices are calculated, this is referred as a truncated SVD. The procedure will calculate the leading columns of these matrices much faster than it can calculate the later columns. Thus smaller values of  $k$  result in faster execution times. The algorithm used in this routine is designed to find these leading columns only. The value of  $k$  (or MAX\_k) should be less than the minimum of the number of rows and number of columns in the original noncompressed matrix. If you attempt to calculate too many columns the calculations might not be accurate and calculation times can be slow (in the former case a warning will be written to the log).

The matrices  $U$  and  $V$  are orthonormal (that is, each column is orthogonal to every other column in the matrix, and the Euclidean norm of each column is 1).  $\Sigma$  is diagonal with monotonically decreasing non-negative real entries. These matrices have many interesting and important qualities. For text mining we are interested in using them for dimension reduction.

To see how we accomplish this, we view the columns of the original matrix  $A$  as coordinates. Each column then specifies the position of a data point (document) in a high dimensional space. Similarly, the first column of  $U$  defines a point in space. If we join this point and the origin, this defines a line. That line is the best least squares fit to the data (distance measured perpendicular to the line). In other words, the first column of  $U$ ,  $u_1$ , minimize

$$\sum_{i=1}^n \left\| a_i - c_i u_1 \right\|$$

where  $a_i$  is the  $i^{\text{th}}$  column of  $A$ ,  $c_i$  is some constant,  $n$  is the number of columns in  $A$ , and  $\|\cdot\|$  is the Euclidean norm. The second column of  $U$ ,  $u_2$ , also defines a line, and together with  $u_1$ , it defines a plane. This plane is the best fit 2-dimensional subspace to the data (again in a least squares sense). Generally, the first  $k$  columns of  $U$  form a best-fit least-square  $k$ -dimensional subspace. We can therefore project the columns of  $A$  onto the first  $k$  columns of  $U$  to reduce the dimension. The result of projecting a document,  $d$ , is  $k$  real numbers,  $p_i$ , each of which is the inner-product of  $d$  onto a column of  $U$ . This projection is defined by  $p_i = d^T u_i$ .

For text mining, PROC SPSVD is generally used to calculate the SVD for the training data set. The training data set is then projected onto the first  $k$  columns of  $U$  as discussed above. The value for  $k$  can be chosen by the user or can be recommended by the procedure. Generally,  $k$  must be large enough to capture much of the meaning in the document collection, but it should not be so large as to capture the noise. Values between 50 and 200 work well for large document collection that contains thousand of documents. Once the training data set has been projected, the matrices used for the projection are saved, and the PROC is invoked again to project the validation and any test or score data sets onto these same matrices. Many options exist for weighting the

input matrix and scaling or normalizing the projected image. These are discussed in the syntax sections.

It should be noted that the first  $k$  columns of  $V$  also form a best fit subspace with respect to the rows of  $A$  (not just for the columns). PROC SPSVD allows the user to project the rows onto the first  $k$  columns of  $V$  for this reason. In the framework of text mining, this is viewed as a representation for the terms in the data set. Possible uses for this include clustering the reduced dimension representation of the terms to find concepts prevalent in the document collection.

---

## Syntax: SPSVD Procedure

```
PROC SPSVD <option(s)>;
  COL variable;
  ENTRY variable;
  OUTPUT <option(s)>;
  ROW variable;
```

---

## PROC SPSVD Statement

Invoke the SPSVD procedure.

```
PROC SPSVD <option(s)>;
```

### Options

**DATA = <libref.>SAS-data-set**

Specifies the data set to be factored. This data set should be in compressed form as described in the overview. If you omit the DATA= option, the procedure uses the most recently created SAS data set.

**GLOBAL = NORMAL | GFIDF | IDF | ENTROPY**

Specifies a global term weight,  $w_i$ , to be used to weight the entries of the input matrix before any calculations. If the WGT = option is specified, the weighted matrix will be written out. Local and global weights are combined so that an entry,  $\hat{f}_{ij}$ , of the new matrix is calculated from an entry,  $f_{ij}$ , of the old matrix as  $\hat{f}_{ij} = w_i * g(f_{ij})$ . If the local weight is not specified, global term weight defaults to  $f_{ij}$ . If a global weight is not specified, cell weight defaults to 1. The GLOBAL option can not be used in conjunction with the IN\_GLOBAL option. The GWGT option on the OUTPUT statement enables you to save the calculated global weights so they can be applied to subsequent data sets by using the IN\_GLOBAL option.

Global weights are functions of the row entries of the original, noncompressed, sparse matrix. The following lists the available global weights:

NORMAL

$$w_i = \frac{1}{\sqrt{\sum_j f_{ij}^2}}$$

GFIDF (Global Frequency Inverse Document Frequency)

$$w_i = \frac{g_i}{d_i}$$

IDF (Inverse Document Frequency)

$$w_i = \log_2 \left( \frac{n}{d_i} \right) + 1$$

ENTROPY

$$w_i = 1 + \sum_j \frac{p_{ij} \log_2(p_{ij})}{\log_2(n)}$$

For the above equations,

- $f_{ij}$  is the frequency of term  $i$  in document  $j$
- $d_i$  is the number of documents in which term  $i$  appears
- $g_i$  is the number of times that term  $i$  appears in the whole document collection
- $n$  is the number of documents in the collection
- 

$$p_{ij} = \frac{f_{ij}}{g_i}$$

**IN\_GLOBAL = <libref.>SAS-data-set**

Specifies the input data set containing global weights. This option is generally used in conjunction with the GWGT options in the output statement. In a predictive modeling setting, we want to calculate global weights based only on the training data set. These weights are then written to a data set using the GWGT option. We can then apply these same weights to another data set. For example, the weights can be applied to the validation data set by using the IN\_GLOBAL option. Note that the IN\_GLOBAL option and the GLOBAL option cannot both be specified.

**IN\_U = <libref.>SAS-data-set**

Specifies a  $U$  matrix to be used for a column projection. If this option is used, the SVD of the input matrix is not calculated. Rather, the specified  $U$  matrix is used for projections.

**IN\_S = <libref.>SAS-data-set**

Specified a  $\Sigma$  matrix to be used for a projection. If this option is used, the SVD of the input matrix is not calculated. Rather, the  $U$ ,  $V$ , and  $\Sigma$  matrices specified are used for projections. Only the matrices needed for the requested projections need to be supplied.

**IN\_V = <libref.>SAS-data-set**

Specifies a  $V$  matrix to be used for a row projection. If this option is used, the SVD of the input matrix is not calculated. Rather, the  $V$  matrix is used for projections. Only the matrices needed for the requested projections need to be supplied.

**k = integer**

Represents the number of dimensions that the data set will be reduced to, and controls the number of columns of  $U$ ,  $\Sigma$ , and  $V$  to be calculated and used for projections. The procedure will calculate the number only as specified. Therefore, specifying a  $k$  larger than is needed will cause the procedure to run for an unnecessary long time. See the Overview section for tips on choosing  $k$ . If  $U$  and  $V$  are passed to the procedure via IN\_U or IN\_V, then the user does not need to specify  $k$  as this will be deduced from the number of columns in the passed data sets.

**LOCAL = BINARY|LOG|NONE**

Specifies a local cell weight to be used to weight the entries of the input matrix before any calculations. If the WGT = option is specified, the weighted matrix will be written out. Local and global weights are combined so that an entry,  $\hat{f}_{ij}$ , of the new matrix is calculated from an entry,  $f_{ij}$ , of the old matrix as  $\hat{f}_{ij} = w_i * g(f_{ij})$ . If the local weight is not specified, cell weight defaults to  $f_{ij}$ . If a global weight is not specified, cell weight defaults to 1.

The following lists the available local weights:

BINARY	If the term occurs in the document, then cell weighing is the indicator function of $f_{ij}$ , that is, it assigns a 1 if $f_{ij}$ is nonzero. Otherwise it leaves the frequency as 0
LOG	$cellweight = \log_2(f_{ij} + 1)$
NONE	No weighting function is applied to the cell.

**MAX\_k = integer**

Specifies the maximum dimension number that the procedure should return as the recommended value. This option is used when you want the procedure to recommend the number of dimensions ( $k$ ). The value of MAX\_k tells the procedure to recommend at most MAX\_k dimensions. MAX\_k also serves as the number of dimensions that the procedure will attempt to calculate (as opposed to recommend). The option is not used if the option k= is specified.

**p = integer**

Specifies the number of iterations, beyond  $k$ , before the procedure restarts. PROC SPSVD is an iterative procedure. As iterations continue, more and more memory is used and the procedure slows down due to number of calculations required. If the desired quantities have not been calculated to acceptable accuracy within  $k+p$  iterations, the procedure will restart, maintaining much of the information learned in the first  $k+p$  iterations. Setting the value of  $p$  low will cause frequent restarts which will use less memory. However, the restarting takes time so this might slow the procedure. Conversely, if  $p$  is too large, the routine will begin to slow due to the calculations required. If this value is not specified, it defaults to  $\min\{k, 75\}$ .

**RES = LOW|MEDIUM|HIGH**

specifies the resolution for recommending the number of dimensions. A low resolution typically recommends fewer dimensions than a high resolution does. This option is ignored when k= options is specified. When RES= option is used, the number recommended SVD dimensions ( $k$ ) is determined heuristically based on the the amount of variance explained by the maximum number of SVD dimensions you specify. Suppose this value is maxdim and these maxdim SVD dimensions accounts for  $p\%$  of the total variance. The setting of High resolution will always recommend the maximum number of SVD dimensions, maxdim. For medium resolution, the recommended number of SVD dimensions is based on the number of dimensions that account for  $5/6 * (p\%$  of the total variance). Finally, for low resolution, the recommended number of SVD dimensions that accounts for  $2/3 * (p\%$  of the total

variance). The second column (entitled KEEP) of the output data set for S indicates the recommended number of dimensions by placing a 1 for rows that should be kept.

**TOL = *number***

Specifies a tolerance for the procedure to stop finding eigenvalues of  $A^T A$ . Suppose  $\theta$  is the eigenvalue estimate and  $y$  is the eigenvector estimate, then the procedure terminates when all  $k$  sets of values satisfy  $\|A^T A y - y \theta\|^2 \leq TOL$ . If TOL is not specified, it defaults to  $10^{-6}$ , which is more than adequate for most text mining problems.

## COL Statement

Specifies the row variable. This statement is not required if the column variable has a name of COL.

**COL** *variable*;

***variable***

Specifies the name of the variable in the input data set that contains the column variable for the compressed matrix format as described in the overview.

## ENTRY Statement

Specifies the variable name of the entry values. This statement is not required if the variable has a name of ENTRY.

**ENTRY** *variable*;

***variable***

Specifies the name of the variable in the input data set that contains the entry values for the compressed matrix format as described in the overview.

## OUTPUT Statement

Specifies the data sets to be output.

**OUTPUT** *<option(s)>*;

### Options

**COLPRO|DOCPRO** = *<libref.>SAS-data-set*

Specifies the data set that the projection of the columns of the input matrix onto the columns of the matrix  $U$  will be written to. If the IN\_U option is specified, the data in the set specified by the IN\_U option will be used for the projection. Otherwise  $U$  will be calculated from the input data set. If SCALECOL|SCALEDOC or SCALEALL is specified and IN\_U is specified, then IN\_S must also be specified.

**GWGT = <libref.>SAS-data-set**

Specifies the name of the data set that contains the calculated global weights. This data set can be applied to other data sets by using the IN\_GLOBAL option. This option must be used in conjunction with the GLOBAL option.

**NORMCOL|NORMDOC, NORROW|NORMWORD, NORMALL**

Requests to normalize the Euclidean length of the document (column), word (row) or both projections. For example, if NORMCOL, NORMDOC, or NORMALL is specified, then each observation in the data set specified by the DOCPRO option will have a length of 1. Similarity between terms or document vectors is typically measured with the angle between the vectors, but this option is useful because once normalized, euclidean distance can be used to compare similarity.

**ROWPRO|WORDPRO = <libref.>SAS-data-set**

Specifies the data set that the projection of the rows of the input matrix onto the rows of the matrix  $V$  will be written to. If the IN\_V option is specified, the data in the set specified by the IN\_V option will be used for the projection. Otherwise  $V$  will be calculated from the input data set. If SCALEROW|SCALEWORD or SCALEALL is specified and IN\_V is specified, then IN\_S must also be specified.

**S = <libref.>SAS-data-set**

Specifies the name of the data set to store the calculated  $\Sigma$  matrix. The matrix is written with rows of the matrix as observations in the SAS data set and columns as variables. The variables are named COL1–COLk. You cannot specify S = if the IN\_S option has been specified.

**SCALECOL|SCALEDOC, SCALEROW|SCALEWORD, SCALEALL**

Requests that the associated projections (column, row, or all) be scaled by the inverse of the singular values. SCALEALL specifies that both the document (column) and the word (row) projections should be scaled. SCALECOL or SCALEDOC specifies that the document (columns of the input matrix) projections be scaled. SCALEROW or SCALEWORD specifies that the term (rows of the input matrix) projections to be scaled. If  $p_{ij}$  is the  $i^{\text{th}}$  coordinate of the projected image of the  $j^{\text{th}}$  document, then scaling replaces the formula  $p_{ij} = a_j^T u_i$  with  $p_{ij} = a_j^T u_i \sigma_i^{-1}$  where  $\sigma_i$  is the  $i^{\text{th}}$  singular value (the  $i^{\text{th}}$  entry on the diagonal of  $\Sigma$ ).

Scaling has two functions. First, it puts more weight on those themes in a document that are uncommon in the document collection. Second, if either the terms or documents, but not both, are scaled, and both are placed in the same space then the terms and documents that are highly associated are more likely to be near each other.

**U = <libref.>SAS-data-set**

Specifies the name of the data set to store the calculated  $U$  matrix. The matrix is written with rows of the matrix as observations in the SAS data set and columns as variables. The variables are named COL1–COLk. You cannot specify U = if the IN\_U option has been specified.

**V = <libref.>SAS-data-set**

Specifies the name of the data set to store the calculated  $V$  matrix. The matrix is written with rows of the matrix as observations in the SAS data set and columns as variables. The variables are named COL1–COLk. You cannot specify V = if the IN\_V option has been specified.

**WGT = <libref.>SAS-data-set**

Specifies the name of the data set to which the procedure writes the weighted matrix, if the LOCAL, GLOBAL, or both statements are used. If LOCAL and /or GLOBAL is specified, but WGT= is not, then all calculations performed by the procedure are still based on the weighted matrix; but the weighted matrix will not be saved. If WGT= is specified and COLPRO, ROWPRO, U=, S=, and V= are not specified, then the matrix will be weighted and written to disk; no other calculations will be performed.

---

## ROW Statement

Specifies the row variable. This statement is not required if the row variable has a name of ROW.

*ROW variable;*

***variable***

Specifies the name of the variable in the input data set that contains the row variable for the compressed matrix format as described in the overview.

---

## Example: SPSVD Procedure

---

### Example 1: PROC SPSVD

Suppose there are two data set, SASUSER.TRAIN, and SASUSER.VALID produced by the *Text Parsing* node in Enterprise Miner. You want to use SASUSER.TRAIN for training a predictive model and the SASUSER.VALID for validation.

```
PROC SPSVD DATA=SASUSER.TRAIN K=200 P=50 LOCAL=LOG GLOBAL=ENTROPY;
  ROW KEY;
  COL DOC;
  ENTRY COUNT;
  OUTPUT U=SASUSER.U V=SASUSER.V S=SASUSER.S NORMALL SCALEALL
         DOCPRO=SASUSER.TRAINDP GWGT=SASUSER.WEIGHTS;
RUN;

PROC SPSVD DATA=SASUSER.VALID IN_U=SASUSER.U IN_S=SASUSER.S
  LOCAL=LOG IN_GLOBAL=SASUSER.WEIGHTS;
  ROW KEY;
  COL DOC;
  ENTRY COUNT;
  OUTPUT NORMALL SCALEALL DOCPRO=SASUSER.VALIDDP;
RUN;
```

The first PROC statement applies a local log, global entropy weighting scheme to the training data set. The ROW, COL, and ENTRY options specify the names given to these variables by the Text Parsing node in Enterprise Miner. Once the procedure has weighted the matrix, it calculates 200 (specified in the K= option) columns of  $U$ ,  $\Sigma$ , and

$V$  based on this weighted matrix. The weighted training data set is then projected onto the first 200 columns of  $U$ , scaled by the inverse singular values and its length is normalized. The result of the projection is written to SASUSER.TRAINDP. The calculated global weights (entropy in this case) are saved to the data set SASUSER.WEIGHTS.

The second PROC statement is to project the validation data set using the calculations from the training data set. This is done by specifying the  $U$  and  $\Sigma$  matrices calculated in the first PROC step with the IN\_U and IN\_S options. Notice that you do not need to specify the V data set since you are not projecting the terms. To project the document in the validation data set, specify the same local weighting option for the validation data set and pass the calculated global weights via the IN\_GLOBAL option. Then, request that the normalized, scaled projection be written to SASUSER.VALIDDP. This way the validation data set is weighted in exactly the same way as the training data set. Using the GLOBAL option on the validation data set would cause new global weights to be calculated based on the data in this set, which is not appropriate in this example because you want each dimension in the validation data set to correspond to a dimension in the training data set.

---

## References

- Berry, M.W. and Browne, M. *Understanding Search Engines: Mathematical Modeling and Text Retrieval*. Siam, Philadelphia, 1999.
- Deerwester, S., Dumais, S.T., and Furnas, G.W., and Landauer, T.K. Indexing by Latent Semantic Analysis. *Journal of the American Society for Information Science*. 41(6): 391–407, 1990.
- Trefethen, L.N. and Bau, D. *Numerical Linear Algebra*. Siam, Philadelphia, 1997.
- Watkins, D.S. *Fundamentals of Matrix Computations*. John Wiley and Sons, New York, 1991.