



Agilent Technologies

ADS 2008
January 2008
MATLAB Blocks

Advanced Design System 2008

© Agilent Technologies, Inc. 2000-2008

395 Page Mill Road, Palo Alto, CA 94304 U.S.A.

No part of this manual may be reproduced in any form or by any means (including electronic storage and retrieval or translation into a foreign language) without prior agreement and written consent from Agilent Technologies, Inc. as governed by United States and international copyright laws.

Acknowledgments

Mentor Graphics is a trademark of Mentor Graphics Corporation in the U.S. and other countries. Microsoft®, Windows®, MS Windows®, Windows NT®, and MS-DOS® are U.S. registered trademarks of Microsoft Corporation. Pentium® is a U.S. registered trademark of Intel Corporation. PostScript® and Acrobat® are trademarks of Adobe Systems Incorporated. UNIX® is a registered trademark of the Open Group. Java™ is a U.S. trademark of Sun Microsystems, Inc. SystemC® is a registered trademark of Open SystemC Initiative, Inc. in the United States and other countries and is used with permission. MATLAB® is a U.S. registered trademark of The Math Works, Inc.. HiSIM2 source code, and all copyrights, trade secrets or other intellectual property rights in and to the source code in its entirety, is owned by Hiroshima University and STARC.

Errata The ADS product may contain references to "HP" or "HPEESOF" such as in file names and directory names. The business entity formerly known as "HP EEsof" is now part of Agilent Technologies and is known as "Agilent EEsof". To avoid broken functionality and to maintain backward compatibility for our customers, we did not change all the names and labels that contain "HP" or "HPEESOF" references.

Warranty The material contained in this document is provided "as is", and is subject to being changed, without notice, in future editions. Further, to the maximum extent permitted by applicable law, Agilent disclaims all warranties, either express or implied, with regard to this manual and any information contained herein, including but not limited to the implied warranties of merchantability and fitness for a particular purpose. Agilent shall not be liable for errors or for incidental or consequential damages in connection with the furnishing, use, or performance of this document or of any information contained herein. Should Agilent and the user have a separate written agreement with warranty terms covering the material in this document that conflict with these terms, the warranty terms in the separate agreement shall control.

Technology Licenses The hardware and/or software described in this document are furnished under a license and may be used or copied only in accordance with the terms of such license. Portions of this product include the SystemC software licensed under Open Source terms, which are available for download at <http://systemc.org/>. This software is redistributed by Agilent. The Contributors of the SystemC software provide this software "as is" and offer no warranty of any kind, express or implied, including without limitation warranties or conditions or title and non-infringement, and implied warranties or conditions merchantability and fitness for a particular purpose. Contributors shall not be liable for any damages of any kind including without limitation direct, indirect, special, incidental and consequential damages, such as lost profits. Any provisions that differ from this disclaimer are offered by Agilent only.

Restricted Rights Legend If software is for use in the performance of a U.S. Government prime contract or subcontract, Software is delivered and licensed as "Commercial computer software" as defined in DFAR 252.227-7014 (June 1995), or as a "commercial item" as defined in FAR 2.101(a) or as "Restricted computer software" as defined in FAR 52.227-19 (June 1987) or any equivalent agency regulation or contract clause. Use, duplication or disclosure of Software is subject to Agilent Technologies' standard commercial license terms, and non-DOD Departments and Agencies of the U.S. Government will receive no greater than Restricted Rights as defined in FAR 52.227-19(c)(1-2) (June 1987). U.S. Government users will receive no greater than Limited Rights as defined in FAR 52.227-14 (June 1987) or DFAR 252.227-7015 (b)(2) (November 1995), as applicable in any technical data.

Contents

- About Numeric Matlab
 - EigCx_M
 - Erf
 - Erfc
 - Matlab_M
 - MatlabCx_M
 - MatlabF_M
 - MatlabFCx_M
 - MatlabLibLink
 - MatlabLibLinkCx
 - MatlabSink
 - MatlabSinkF
 - Norm_M
 - RandDeintrlv
 - RandIntrlv
 - SVD_Cx_M

About Numeric Matlab

The MATLAB models provide an interface between ADS Ptolemy and MATLAB, a numeric computation and visualization environment from The MathWorks, Inc. All ADS Ptolemy MATLAB Cosimulation models can be found in the Numeric Matlab component palette and library on the Digital Signal Processing (DSP) Schematic. There are three types of MATLAB models that are built into ADS Ptolemy:

- Models that interpret a MATLAB script - The script can contain a function, command, statement or several statements.
 - [MatlabCx_M](#)
 - [MatlabFCx_M](#)
 - [MatlabF_M](#)
 - [MatlabSink](#)
 - [MatlabSinkF](#)
 - [Matlab_M](#)
- Models that can compile and import a MATLAB function as a shared library - The function can be implemented in a .m file or a pre-compiled shared library.
 - [MatlabLibLink](#)
 - [MatlabLibLinkCx](#)
- Models that call specific MATLAB built-in functions
 - [EigCx_M](#)
 - [Norm_M](#)
 - [RandDeintrlv](#)
 - [RandIntrlv](#)
 - [SVD_Cx_M](#)

- [Erf](#)
- [Erfc](#)

The components shown in this last section above represent examples of built-in MATLAB functions that are imported for Advanced Design System (ADS) DSP simulation. These imported models demonstrate different methods for interfacing MATLAB blocks with other ADS models. Each model has relevance in communications system analysis and design.

While the import process to construct such models can be performed manually, these models have been imported using the Import MATLAB Function Tool included as part of ADS. For more information about this tool, refer to [Importing MATLAB Functions](#).

Note
 Use of these models requires that you have MATLAB properly installed and configured for ADS co-simulation. For more information on MATLAB and details on the MATLAB implementation of the various computations used in this document, visit The MathWorks web site at <http://www.mathworks.com>.

EigCx_M



Description Eigenvalue Decomposition of General Complex Matrix
 Library Numeric Matlab

Pin Inputs

Pin	Name	Description	Signal Type
1	X	Input matrix	complex matrix

Pin Outputs

Pin	Name	Description	Signal Type
2	V	Eigenvectors	complex matrix
3	D	Eigenvalues	complex matrix

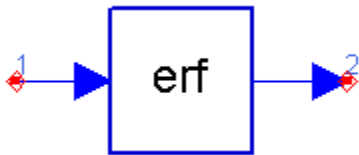
Notes/Equations

1. This model performs the eigenvalue decomposition of a complex matrix using the MATLAB eig function, $[V, D]=$

$\text{eig}(A)$. For details on the MATLAB implementation of this computation, visit The MathWorks website at <http://www.mathworks.com>.

- The eigenvalue decomposition takes a complex matrix on the input (pin 1) and outputs a complex matrix whose columns represent the eigenvectors of the input (pin 2) and a diagonal matrix whose complex entries represent the eigenvalues of the input (pin 3).
- In communication system and signal processing analysis, the eigenvalue decomposition is useful for determining the important subspace of a matrix entity. For example, suppose A represents a signal covariance matrix. If we let D_0 represent D with all entries whose magnitude is smaller than a threshold set to zero, then VD_0V^{-1} represents an efficient approximation to the covariance. Typically for a covariance, $V^{-1} = V^H$, where the superscript H represents a conjugate transpose. In this case, only a subset of the columns of V (the subspace) is used to approximate the covariance.

Erf



Description Error Function
Library Numeric Matlab

Pin Inputs

Pin	Name	Description	Signal Type
1	x	Input data	real

Pin Outputs

Pin	Name	Description	Signal Type
2	y	Output data	real

Notes/Equations

- This model computes the error function of the input data using the MATLAB erf function, $[y] = \text{erf}(x)$. For details on the MATLAB implementation of this computation, visit The MathWorks website at <http://www.mathworks.com>.

- The error function is mathematically represented as:

$$y = \text{erf}(x) = \frac{2}{\sqrt{\pi}} \int_0^x \exp(-t^2) dt$$

- The error function is commonly used on communication system analysis to determine the probability of correctly

detecting a symbol when the additive noise is described as a Gaussian random process.

Erfc



Description Complementary Error Function
Library Numeric Matlab

Pin Inputs

Pin	Name	Description	Signal Type
1	x	Input data	real

Pin Outputs

Pin	Name	Description	Signal Type
2	y	Output data	real

Notes/Equations

1. This model computes the complementary error function of the input data using the MATLAB erfc function, $[y] = \text{erfc}(x)$. For details on the MATLAB implementation of this computation, visit The MathWorks website at <http://www.mathworks.com>.
2. The complementary error function is mathematically represented as:

$$y = \text{erfc}(x) = \frac{2}{\sqrt{\pi}} \int_x^{\infty} \exp(-t^2) dt$$

3. The complementary error function is commonly used on communication system analysis to determine the probability of correctly detecting a symbol when the additive noise is described as a Gaussian random process. In fact, the Q function commonly used in bit error rate analysis is related to the complementary error function using:

$$Q(x) = \frac{1}{2} \text{erfc}\left(\frac{x}{\sqrt{2}}\right)$$

Matlab_M



Description Matlab Floating Point Matrix Output

Library Numeric, Matlab

Class SDFMatlab_M

Derived From Matlab

Parameters

Name	Description	Default	Type
ScriptDirectory	An optional directory specifying where to find any custom Matlab files.		string
MatlabSetUp	Matlab command to execute during begin method		string
MatlabFunction	Matlab command to execute for each simulation sample		string
MatlabWrapUp	Matlab command to execute during wrapup method		string

Pin Inputs

Pin	Name	Description	Signal Type
1	input		multiple anytype

Pin Outputs

Pin	Name	Description	Signal Type
2	output		multiple real matrix

Notes/Equations

1. Matlab_M evaluates Matlab functions on its inputs and outputs floating-point (real) matrices.

2. ScriptDirectory is an optional directory specifying where to find any custom Matlab files.
3. The Matlab_M component has parameters called MatlabSetUp and MatlabWrapUp as shown in the parameter table above. These parameters can refer to a *.m file.
4. For more information about Matlab components, refer to [Introduction to MATLAB Cosimulation](#).
5. For more information regarding numeric matrix component signals, refer to [Numeric Matrix Components](#).

MatlabCx_M



Description Matlab Complex Matrix Output
 Library Numeric, Matlab
 Class SDFMatlabCx_M
 Derived From Matlab_M

Parameters

Name	Description	Default	Type
ScriptDirectory	An optional directory specifying where to find any custom Matlab files.		string
MatlabSetUp	Matlab command to execute during begin method		string
MatlabFunction	Matlab command to execute for each simulation sample		string
MatlabWrapUp	Matlab command to execute during wrapup method		string

Pin Inputs

Pin	Name	Description	Signal Type
1	input		multiple anytype

Pin Outputs

Pin	Name	Description	Signal Type
2	output		multiple complex matrix

Notes/Equations

1. MatlabCx_M evaluates Matlab functions on its inputs and outputs complex matrices.
2. ScriptDirectory is an optional directory specifying where to find any custom Matlab files.
3. For more information about Matlab components, refer to [Introduction to MATLAB Cosimulation](#).
4. For more information regarding numeric matrix component signals, refer to [Numeric Matrix Components](#).

MatlabF_M



Description Matlab Floating Point Matrix Output with Scripts Importing
 Library Numeric, Matlab
 Class SDFMatlabF_M
 Derived From Matlab

Parameters

Name	Description	Default	Type
ScriptDirectory	An optional directory specifying where to find any custom Matlab files.		string
MatlabSetUp	Matlab command to execute during begin method		filename
MatlabFunction	Matlab command to execute for each simulation sample		filename
MatlabWrapUp	Matlab command to execute during wrapup method		filename

Pin Inputs

Pin	Name	Description	Signal Type
1	input		multiple anytype

Pin Outputs

Pin	Name	Description	Signal Type
2	output		multiple real matrix

Notes/Equations

1. MatlabF_M evaluates Matlab functions on its inputs and outputs floating-point (real) matrices.
2. MatlabSetup, MatlabFunction, and MatlabWrapUp inputs accept script files only.
3. ScriptDirectory is an optional directory specifying where to find any custom Matlab files referenced inside MatlabSetup, MatlabFunction, and MatlabWrapUp scripts.
4. The MatlabF_M component has parameters called MatlabSetUp and MatlabWrapUp as shown in the parameter table above. These parameters can refer to a *.m file.
5. For more information about Matlab components, refer to [Introduction to MATLAB Cosimulation](#).
6. For more information regarding numeric matrix component signals, refer to [Numeric Matrix Components](#).

MatlabFCx_M



Description Matlab Complex Matrix Output with Scripts Importing
 Library Numeric, Matlab
 Class SDFMatlabFCx_M
 Derived From MatlabF_M

Parameters

Name	Description	Default	Type
ScriptDirectory	An optional directory specifying where to find any custom Matlab files.		string
MatlabSetUp	Matlab command to execute during begin method		filename

MatlabFunction	Matlab command to execute for each simulation sample		filename
MatlabWrapUp	Matlab command to execute during wrapup method		filename

Pin Inputs

Pin	Name	Description	Signal Type
1	input		multiple anytype

Pin Outputs

Pin	Name	Description	Signal Type
2	output		multiple complex matrix

Notes/Equations

1. MatlabFCx_M evaluates Matlab functions on its inputs and outputs complex matrices.
2. MatlabSetup, MatlabFunction, and MatlabWrapUp inputs accept script files only.
3. ScriptDirectory is an optional directory specifying where to find any custom Matlab files referenced inside MatlabSetup, MatlabFunction, and MatlabWrapUp scripts.
4. For more information about Matlab components, refer to [Introduction to MATLAB Cosimulation](#).
5. For more information regarding numeric matrix component signals, refer to [Numeric Matrix Components](#).

MatlabLibLink



Description Matlab Cosimulation with Compile Mode Support
 Library Numeric, Matlab
 Class SDFMatlabLibLink

Parameters

Name	Description	Default	Type
------	-------------	---------	------

Advanced Design System 2008

Library	Compiled Matlab Library name, must start with lib		string
Setup	Matlab Setup function name		filename
SetupParm	Optional Setup Parameter list		complex array
Function	Matlab function name to be executed for each simulation sample		filename
Mode	Mode of operation, automatic, force recompile or force script running: AUTO, COMPILE, SCRIPT	AUTO	enum

Pin Inputs

Pin	Name	Description	Signal Type
1	input		multiple anytype

Pin Outputs

Pin	Name	Description	Signal Type
2	output		multiple real matrix

Notes/Equations

1. Ensure Matlab and Matlab Compiler are properly installed; ADS will call the compiler for you; makefiles or manual compilation is not needed.
2. The Library name must start with lib; for example, libfoo, libbar. It specifies the library name to be compiled or the library that already exists to link in. Library will be compiled into `_prj \ data` directory. Pre-compiled shared library can also be placed under `_prj \ data` directory to be picked up.
3. SetupParm specifies a list of function arguments that are passed into Setup Matlab function. The exact number of arguments specified in the given setup matlab file must be provided; otherwise the simulation will crash without any warnings.
The format is comma-separated arguments in curly braces `{}`; for example, `{a,b,c}`.
 - a, b, c can be ADS VAR variables, constants or equations.
 - `{a,b,c}` on the schematic must not be surrounded by quotes; the component dialog box format is `@{a,b,c}`.
4. Setup is the function name or . m file. You can browse Setup and Function .m files. It can remain blank if setup is not needed. The Library will be compiled under `your_prj \ data` directory.
5. Arguments of function are data passed from Input and Output Ports of the components. Input Port 1 will be mapped to Input Argument 1.
6. Mode options are:
 - AUTO will ONLY compile the .m files if Library with specified name is not found in the library search path, LD_LIBRARY_PATH on Sun, SHLIB_PATH on Unix, etc.

- COMPILE will always recompile the .m files.
- SCRIPT will use the traditional Matlab link; it simply interprets .m files.

MatlabLibLinkCx



Description Matlab Cosimulation with Compile Mode Support and Complex Output
 Library Numeric, Matlab
 Class SDFMatlabLibLinkCx
 Derived From MatlabLibLink

Parameters

Name	Description	Default	Type
Library	Compiled Matlab Library name, must start with lib		string
Setup	Matlab Setup function name		filename
SetupParm	Optional Setup Parameter list		complex array
Function	Matlab function name to be executed for each simulation sample		filename
Mode	Mode of operation, automatic, force recompile or force script running: AUTO, COMPILE, SCRIPT	AUTO	enum

Pin Inputs

Pin	Name	Description	Signal Type
1	input		multiple anytype

Pin Outputs

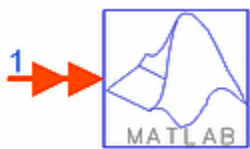
Pin	Name	Description	Signal Type
-----	------	-------------	-------------

2	output		multiple complex matrix
---	--------	--	-------------------------

Notes/Equations

1. Ensure Matlab and Matlab Compiler are properly installed; ADS will call the compiler for you; makefiles or manual compilation is not needed.
2. The Library name must start with lib; for example, libfoo, libbar. It specifies the library name to be compiled or the library that already exists to link in. Library will be compiled into _prj\data directory. Pre-compiled shared library can also be placed under _prj\data directory to be picked up.
3. SetupParm specifies a list of function arguments that are passed into Setup Matlab function. The exact number of arguments specified in the given setup matlab file must be provided; otherwise the simulation will crash without any warnings.
The format is comma-separated arguments in curly braces {}; for example, {a,b,c}.
 - a, b, c can be ADS VAR variables, constants or equations.
 - {a,b,c} on the schematic must not be surrounded by quotes; the component dialog box format is @{a,b,c}.
4. Setup is the function name or .m file. You can browse Setup and Function .m files. It can remain blank if setup is not needed. The Library will be compiled under your _prj\data directory.
5. Arguments of function are data passed from Input and Output Ports of the components. Input Port 1 will be mapped to Input Argument 1.
6. Mode options are:
 - AUTO will ONLY compile the .m files if Library with specified name is not found in the library search path, LD_LIBRARY_PATH on Sun, SHLIB_PATH on Unix, etc.
 - COMPILE will always recompile the .m files.
 - SCRIPT will use the traditional Matlab link; it simply interprets .m files.

MatlabSink



Description Matlab Function
 Library Numeric, Matlab
 Class SDFMatlabSink
 Derived From Matlab_M

Parameters

Name	Description	Default	Type	Range
ScriptDirectory	An optional directory specifying where to find any custom		string	

	Matlab files.			
MatlabSetUp	Matlab command to execute during begin method		string	
MatlabFunction	Matlab command to execute for each simulation sample		string	
MatlabWrapUp	Matlab command to execute during wrapup method		string	
NumberOfFirings	number of invocations during a simulation	1	int	[1, ∞)

Pin Inputs

Pin	Name	Description	Signal Type
1	input		multiple anytype

Notes/Equations

1. MatlabSink behaves like other Matlab components except that it does not have any output. The total amount of data collected is determined by NumberOfFirings.
2. ScriptDirectory is an optional directory specifying where to find any custom Matlab files.
3. For more information about Matlab components, refer to [Introduction to MATLAB Cosimulation](#).
4. For more information regarding numeric matrix component signals, refer to [Numeric Matrix Components](#).

MatlabSinkF



Description Matlab Function with Scripts Importing
 Library Numeric, Matlab
 Class SDFMatlabSinkF
 Derived From MatlabF_M

Parameters

Name	Description	Default	Type	Range
ScriptDirectory	An optional directory specifying where to find any custom Matlab files.		string	
MatlabSetUp	Matlab command to execute during begin method		filename	
MatlabFunction	Matlab command to execute for each simulation sample		filename	
MatlabWrapUp	Matlab command to execute during wrapup method		filename	
NumberOfFirings	number of invocations during a simulation	1	int	[1, ∞)

Pin Inputs

Pin	Name	Description	Signal Type
1	input		multiple anytype

Notes/Equations

1. MatlabSinkF behaves like other Matlab components except that it does not have any output. The total amount of data collected is determined by NumberOfFirings.
2. MatlabSetup, MatlabFunction, and MatlabWrapUp inputs accept script files only.
3. ScriptDirectory is an optional directory specifying where to find any custom Matlab files referenced inside MatlabSetup, MatlabFunction, and MatlabWrapUp scripts.
4. For more information about Matlab components, refer to [Introduction to MATLAB Cosimulation](#).
5. For more information regarding numeric matrix component signals, refer to [Numeric Matrix Components](#).

Norm_M



Description Matrix Norm

Library Numeric Matlab

parameters

Name	Description	Default	Type
p	Indicates whether to use a 1-norm or 2-norm	two_norm	real enum

Pin Inputs

Pin	Name	Description	Signal Type
1	A	Input matrix	complex matrix

Pin Outputs

Pin	Name	Description	Signal Type
2	n	Matrix norm	real

Notes/Equations

1. This model computes the norm of the input matrix using the MATLAB norm function, $[n]=\text{norm}(A, p)$. For details on the MATLAB implementation of this computation, visit The MathWorks website at <http://www.mathworks.com>.
2. The 1-norm takes the sum of the absolute values of the elements in each column, and returns the maximum over the columns. The 2-norm returns the largest singular value of the input matrix.
3. The matrix norm provides a measure of the magnitude of the matrix elements, and therefore gives an indication of the energy in a matrix (or vector) of signals.

RandDeintrlv



Description Deinterleave Data Interleaved Using Random Permutation

Library Numeric Matlab

Parameters

Name	Description	Default	Type
state	State of the MATLAB random number generator	1234	int
BlockSize	Block size of the interleaved data	128	int

Pin Inputs

Pin	Name	Description	Signal Type
1	idata	Interleaved input data	real

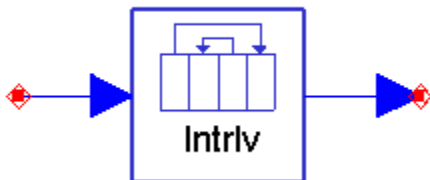
Pin Outputs

Pin	Name	Description	Signal Type
2	data	Deinterleaved data	real

Notes/Equations

1. This model performs deinterleaving of the data at the input using the MATLAB `randdeintrlv` function, `[data] = randdeintrlv(idata, state)`. It requires a license to the Communications Toolbox. For details on the MATLAB implementation of this computation, visit The MathWorks website at <http://www.mathworks.com>.
2. This function takes randomly arranged data presented on the input (pin 1) and presents a rearranged version of the data on the output (pin 2).
3. The interleaving is repeatable if the value of the state parameter is unchanged.
4. If the input interleaved data was created with the `randintrlv` model, then this model can reconstruct the original sequence if it is called with the same value of the state and `BlockSize` parameters.

RandIntrlv



Description Interleave Input Data Stream Using Random Permutation
 Library Numeric Matlab

Parameters

Name	Description	Default	Type
state	State of the MATLAB random number generator	1234	int
BlockSize	Block size of the data to interleave	128	int

Pin Inputs

Pin	Name	Description	Signal Type
1	data	Input data stream	real

Pin Outputs

Pin	Name	Description	Signal Type
2	idata	Interleaved data stream	real

Notes/Equations

1. This model performs interleaving of the data at the input using the MATLAB `randintrlv` function, `[idata] = randintrlv(data, state)`. It requires a license to the Communications Toolbox. For details on the MATLAB implementation of this computation, visit The MathWorks website at <http://www.mathworks.com>.
2. This function takes data presented on the input (pin 1) and presents a randomly rearranged version of the data on the output (pin 2).
3. The interleaving is repeatable if the value of the state parameter is unchanged.
4. To deinterleave the data, use the `randdeintrlv` model with the same value of the state and `BlockSize` parameters.
5. Interleaving is used to provide robustness against burst errors. If a burst of interference results in the loss of several sequential data symbols, it becomes difficult to recover the lost data even when error control coding is applied. However, if the data is rearranged, the lost symbols are interspersed in the data stream, resulting in a scenario in which error control coding will be more effective in recovering the lost data.

SVD_Cx_M



Description Singular value decomposition of a general matrix
 Library Numeric Matlab

Pin Inputs

Advanced Design System 2008

Pin	Name	Description	Signal Type
1	X	Input matrix	complex matrix

Pin Outputs

Pin	Name	Description	Signal Type
2	L	Left singular vectors	complex matrix
3	S	Singular values	real matrix
4	R	Right singular vectors	complex matrix

Notes/Equations

1. This model performs the singular value decomposition of a complex matrix using the MATLAB `svd` function, $[R, S, L] = \text{svd}(X)$. For details on the MATLAB implementation of this computation, visit The MathWorks website at <http://www.mathworks.com>.
2. The singular value decomposition takes a complex matrix on the input (pin 1) and outputs two complex matrices whose columns represent the left and right singular vectors of the input (pins 2 and 4) and a diagonal matrix whose real entries represent the singular vectors of the input (pin 3). It can be applied to any matrix, whether or not it is square.
3. In communication system and signal processing analysis, the singular value decomposition is useful for determining the important subspace of a general matrix entity. The singular vectors represent a basis for the matrix, while the singular values indicate the relative importance of each basis vector.