Tabla de contenido

[Cómo usar Universe-cmd 2](#_Toc485858229)

[Requisito previo 2](#_Toc485858230)

[Ejecución 2](#_Toc485858231)

[Tareas 4](#_Toc485858232)

[Evaluación 4](#_Toc485858233)

[Descubrimiento 4](#_Toc485858234)

[Inferencia 4](#_Toc485858235)

[Breve reseña de algunos tipos de datos *edn* utilizados en los archivos de parámetros. 5](#_Toc485858236)

[cadenas de caracteres (strings) 5](#_Toc485858237)

[símbolos (symbols) 5](#_Toc485858238)

[comentarios (comments) 5](#_Toc485858239)

[listas (lists) 5](#_Toc485858240)

[vectores (vectors) 5](#_Toc485858241)

[mapas (maps) 5](#_Toc485858242)

[Representación de la lógica difusa. 6](#_Toc485858243)

[Tipos de lógicas difusas 6](#_Toc485858244)

[Operadores lógicos (conectivas lógicas) 7](#_Toc485858245)

[Operadores de orden 7](#_Toc485858246)

[Funciones de pertenencia 8](#_Toc485858247)

[Funciones de pertenencia de parámetros fijos. 8](#_Toc485858248)

[Funciones de pertenencia continuas de parámetros variables. 8](#_Toc485858249)

[Estados lingüísticos 9](#_Toc485858250)

[Predicados difusos 9](#_Toc485858251)

[Estructura de los Archivos de parámetros. 10](#_Toc485858252)

[Parámetros de la tarea Evaluación 10](#_Toc485858253)

[Parámetros de la tarea de Descubrimiento 12](#_Toc485858254)

[Generador de predicados difusos 13](#_Toc485858255)

[Predicado simbólico 13](#_Toc485858256)

[Generador de predicados difusos con el predicado simbólico "\*" 13](#_Toc485858257)

[Generador de predicados difusos utilizando asteriscos en el predicado simbólico 14](#_Toc485858258)

[Generador de predicados difusos sin utilizar asteriscos en el predicado simbólico 15](#_Toc485858259)

[Ajuste de los parámetros de las funciones de pertenencia variables. 16](#_Toc485858260)

[Parámetros de la tarea de Inferencia 16](#_Toc485858261)

[Método a utilizar para realizar la inferencia de variables de decisión (llave :method) 18](#_Toc485858262)

[Método "EQV" 18](#_Toc485858263)

[Método "IMP" 18](#_Toc485858264)

[Anexo 1. Funciones de Pertenencia. 19](#_Toc485858265)

[Funciones de pertenencia discretas 19](#_Toc485858266)

[Funciones de pertenencia continuas 20](#_Toc485858267)

# Cómo usar Universe-cmd

## Requisito previo

Antes de ejecutar el programa asegúrese de tener instalado el Java (Java SE Runtime Environment) el cual puede ser descargado desde el siguiente enlace: <http://www.oracle.com/technetwork/java/javase/downloads/jre8-downloads-2133155.html>

## Ejecución

El programa ***universe-cmd*** viene acompañado de dos carpetas contenedoras de los datos y demás archivos de ejemplo; una contiene los sets de datos Excel y la otra los *archivos de parámetros* donde se especifican las tareas a realizar y sus parámetros correspondientes.

**Archivos de Parámetros**

Los archivos de parámetros son ficheros *.txt* escritos en formato **EDN** (Extensible Data Notation), es un formato de datos similar a *JSON*.

Encuentre más información acerca de este formato en el siguiente enlace: <https://github.com/edn-format/edn>

***universe-cmd*** debe ejecutarse desde una terminal o línea de comandos y se le pasará como parámetro la ruta del *archivo de parámetros* a ejecutar.

**Para abrir consola de comandos de Windows**

1. Haga clic en el botón de Inicio.
2. Haz clic en la opción Run (Ejecutar).
3. Escribe en el recuadro ***cmd*** y haz clic en OK para abrir la consola.
4. Utilice el comando *cd* para situarse en la ruta donde se encuentra universe-cmd.exe

**Para abrir terminal en Mac OS X**

1. Haz clic en el icono "Finder" situado en el Dock.
2. Selecciona "Aplicaciones > Utilidades"
3. Finalmente haz doble clic al icono "Terminal"
4. Utilice el comando ***cd*** para situarse en la ruta donde se encuentra universe-cmd

Ejemplo de cómo ejecutar ***universe-cmd.exe*** desde la línea de comandos en windows:

universe-cmd.exe examples\evaluate.txt

Ejemplos de cómo ejecutar ***universe-cmd*** desde la terminal de Mac OS X:

universe-cmd examples/evaluate.txt

./universe-cmd examples/evaluate.txt

En los ejemplos anteriores se le ha pedido a ***universe-cmd*** que ejecute la tarea especificada en el archivo *evaluate.txt* ubicado en la carpeta *examples*.

## Tareas

***universe-cmd*** puede realizar las siguientes tareas:

### Evaluación

La tarea de evaluación calcula los valores de verdad de un predicado difuso de un set de datos.

### Descubrimiento

La terea de descubrimiento busca relaciones entre los estados lingüísticos de un set de datos (predicados difusos) que cumplan con las especificaciones del usuario.

La búsqueda se realiza mediante algoritmos genéticos para la búsqueda de predicados difusos y para el ajuste de los parámetros de las funciones de pertenencia definidas en los estados lingüísticos.

### Inferencia

La tarea de inferencia realiza primeramente una tarea de descubrimiento en un set de datos en el que se han definido los estados lingüísticos de variables de condición y decisión. Los predicados difusos obtenidos son utilizados para inferir los valores de las variables de decisión a partir de otro set de datos en el que sólo se conocen las variables de condición.

# Breve reseña de algunos tipos de datos *edn* utilizados en los archivos de parámetros.

### cadenas de caracteres (strings)

Se representan entre comillas **""**

*ej. "calidad alta"*

### símbolos (symbols)

Los símbolos se utilizan para representar identificadores.

En los archivos de parámetros se pueden utilizar para referirse a las funciones de pertenencia y a los operadores lógicos.

### comentarios (comments)

Texto antecedido por punto y coma **;**

*ej. ; esto es un comentario*

### listas (lists)

Una lista es una secuencia de valores. Las listas pueden tener 0 o más elementos de cualquier tipo. Se representa entre **().** Los elementos dentro de una lista pueden separarse con espacio o con coma.

ejs. ()

(1.1, 1.2)

(1 "uno" 2 (3 4))

### vectores (vectors)

Un vector es una secuencia de valores. Los vectores pueden tener 0 o más elementos de cualquier tipo. Se representa entre **[].** Los elementos dentro de un vector pueden separarse con espacio o con coma.

ejs. []

[1.1, 1.2]

[1 "uno" 2 [3 4]]

### mapas (maps)

Secuencia de asociaciones llave y valor, cada llave solo puede aparecer una vez en el mapa. Los pares llave - valor pueden estar en cualquier orden.

Se representan entre {}. Las llaves están antecedidas por dos puntos ***:*** y los valores pueden ser de cualquier tipo. Los elementos dentro de un mapa pueden separarse con espacio o con coma.

ej.1

{:llave1 "valor 1" :llave2 2 :otrallave {:anidada "si"}}

ej.2

{:llave2 2 :otrallave {:anidada "si"} :llave1 "valor 1"}

Los mapas de los dos ejemplos anteriores son idénticos, aunque se hayan cambiado las posiciones de los pares llave-valor.

Para ganar en legibilidad el mapa anterior se pudiera representar de las siguientes maneras:

Separar los pares llave-valor con coma

{:llave1 "valor 1"**,** :llave2 2**,** :otrallave {:anidada "si"}}

o mejor separándolos por salto de línea

{

:llave1 "valor 1"

:llave2 2

:otrallave {

:anidada "si"

}

}

# Representación de la lógica difusa.

## Tipos de lógicas difusas

*universe-cmd* permite el trabajo con los siguientes tipos de lógicas:

1. Lógica de Zadeh.
2. Lógica Probabilística.
3. Lógica Compensatoria Basada en la Media Geométrica.
4. Lógica Compensatoria Basada en la Media Aritmética.
5. Lógica Compensatoria Basada en la Media Quasi-Aritmética.
6. Lógica Compensatoria Arquimediana.

Los tipos de lógicas se representan mediante un vector que contiene una llave que especifica el tipo de lógica:

|  |  |
| --- | --- |
| Tipo de lógica | Representación |
| Lógica de Zadeh | [:Zadeh] |
| Lógica Probabilística | [:Prob] |
| Lógica Compensatoria basada en la media geométrica | [:GMBC] |
| Lógica Compensatoria basada en la media aritmética | [:AMBC] |

Las lógicas Quasi-Aritmética y la Arquimediana llevan un parámetro extra que indica el valor del exponente, el exponente es un número negativo.

A continuación, se ejemplifica la representación de éstas lógicas tomando el exponente igual a -1.

|  |  |  |
| --- | --- | --- |
| Tipo de lógica | exponente | Representación |
| Lógica Compensatoria Basada en la Media Quasi-Aritmética | -1 | [:QAMBC -1] |
| Lógica Compensatoria Arquimediana | -1 | [:Archimedian -1] |

## Operadores lógicos (conectivas lógicas)

|  |  |  |  |
| --- | --- | --- | --- |
| Operador | Representación | Aridad | Detalle |
| Conjunción (∧) | AND | 2 o más | Depende del tipo de lógica. |
| Disyunción () | OR | 2 o más | Depende del tipo de lógica. |
| Negación () | NOT | 1 (unario) | No depende del tipo de lógica. |
| Implicación (→) | IMP | 2 (binario) | Depende del tipo de lógica. |
| Equivalencia (↔) | EQV | 2 (binario) | Depende del tipo de lógica. |

### Operadores de orden

|  |  |  |  |
| --- | --- | --- | --- |
| Operador | Representación | Aridad | Detalle |
| Orden  estricto | O | 2 (binario) | No depende del tipo de lógica. |
| Orden débil  tipo 1 | OW1 | 2 (binario) | Depende del tipo de lógica. |
| Orden débil  tipo 2 | OW2 | 2 (binario) | Depende del tipo de lógica. |

La aridad es la cantidad de parámetros que puede admitir cada operador para conformar los [predicados difusos](#_Predicados_difusos_2).

## 

## Funciones de pertenencia

Se representan mediante un vector que contiene el nombre de la función de pertenencia y sus parámetros respectivos. Las funciones de pertenencia pueden ser discretas o continuas. En el [Anexo 1](#_Anexo_1._Funciones) se muestran todas las funciones de pertenencia, sus parámetros y gráficas correspondientes.

Las funciones de pertenencia pueden tener parámetros fijos o variables.

### Funciones de pertenencia de parámetros fijos.

Como su nombre los parámetros de éstas funciones de pertenencia permanecen fijos (no varían).

La tarea de Evaluación sólo permite trabajar con este tipo de funciones de pertenencia.

ej. Función de pertenencia gaussiana de parámetros fijos:

*centre* = 5,

*deviation* = 2

[gaussian 5 2]

### Funciones de pertenencia continuas de parámetros variables.

Las funciones de pertenencia continuas de parámetros variables son funciones en las que sus parámetros pueden variar en un intervalo dado. El intervalo de variación de un parámetros es un intervalo semiabierto *[a,b)* (cerrado en *a*, abierto en *b*).

Este tipo de funciones de pertenencia sólo se pueden utilizar en las tareas de Descubrimiento e Inferencia.

En una misma función de pertenencia continua se pueden definir parámetros fijos y variables.

ej.1 Función de pertenencia gaussiana con centro fijo y desviación variable:

*centre* = 5,

1 ≤ *deviation* < 2

[gaussian 5 [1 2]]

ej.2 Función de pertenencia gaussiana con centro y desviación variable:

4.5 ≤ *centre* < 5,

1 ≤ *deviation* < 2

[gaussian [4.5 5] [1 2]]

## Estados lingüísticos

Los estados lingüísticos son una asociación entre el nombre de una columna del set de datos, una función de pertenencia y un nombre de estado o label.

Los estados se representan con un mapa con las siguientes llaves:

|  |  |  |
| --- | --- | --- |
| Llave | Valor | Descripción |
| :label | cadena de caracteres | Nombre del estado lingüístico. Si se omite esta llave se tomará el valor de la llave :colname |
| :colname | cadena de caracteres | Nombre de la columna (header) del set de datos. |
| :f | mapa | Función de pertenencia |

En las tareas de Descubrimiento y de Inferencia se pueden definir "estados linguísticos" sin especificar la función de pertenencia de dichos estados. En este caso *universe-cmd* utilizará Funciones de Pertenencias Generalizadas (FPGs) con parámetros variables.

ej. Definición del estado lingüístico "Temperatura alta" utilizando una función de pertenencia Ltrapezoidal.

{

:label "Temperatura alta"

:colname "temperatura"

:f [Ltrapezoidal 25 30]

}

## Predicados difusos

Los predicados difusos se representan mediante [expresiones S](https://es.wikipedia.org/wiki/Expresi%C3%B3n_S) .

En *universe-cmd* los predicados difusos pueden ser listas o vectores simples o anidados donde el primer elemento de cada lista o vector es un operador lógico y el resto de los elementos pueden ser referencias a estados lingüísticos o bien otros vectores o listas con la misma conformación.

Tenga en cuenta la aridad de cada operador lógico para la conformación de los predicados difusos.

ej.1 Representación del predicado:

*estado1* **Implica** *estado2* (*estado1* ***→*** *estado2)*

Representación en forma de vector:

[**IMP** "estado1" "estado2"]

Representación en forma de lista:

(**IMP** "estado1" "estado2")

La diferencia entre ambas representaciones es el uso de corchetes *[]* o de paréntesis *()*

ej.2 Representación del predicado:

(*estado1* **Y** *estado2* **Y** estado 3) **O** (**No** *estado4)* **Equivale** a (*estado5* **Y** *estado6)*

*(estado1* **∧** *estado2* **∧** *estado3)*  **∨** **¬***estado4* ↔*(estado5* **∧** *estado6)*

Representación en forma de lista:

(**EQV**

(**OR**

(**AND** "estado1" "estado2" "estado3")

(**NOT** "estado4"))

(**AND** "estado5" "estado6"))

# Estructura de los Archivos de parámetros.

Las tareas en universe-cmd se especifican en archivos de parámetros, éstos contendrán datos estructurados edn con toda la información referente a la tarea a realizar.

Cada archivo de parámetros estará compuesto por un mapa con las llaves *:job* y *:query*

**llave *:job***

El valor de la llave *:job* define el tipo de tarea a ejecutar y puede tomar uno de los siguientes valores:

:evaluation

:discovery

:inference

**llave *:query***

El valor de la llave *:query* es un mapa con una estructura que dependerá del tipo de tarea seleccionado.

## Parámetros de la tarea Evaluación

Estructura general de una tarea de Evaluación

{

:job :evaluation

:query

{

; parámetros de la tarea de Evaluación

}

}

El valor de la llave *:query* es un mapa que contiene las siguientes llaves:

|  |  |  |
| --- | --- | --- |
| Llave | Valor | Descripción |
| :db-uri | cadena de caracteres | Ruta del set de datos Excel |
| :out-file | cadena de caracteres | Ruta del archivo de resultados Excel |
| :states | vector de mapas | Estados lingüísticos |
| :logic | vector | Tipo de lógica difusa a utilizar |
| :predicate | vector | Predicado difuso a evaluar |

Ejemplo de archivo de parámetros para realizar una tarea de evaluación:

{

:job :evaluation

:query {

:db-uri "datasets/tinto.xls"

:out-file "evaluation-results.xls"

:states [

;estado lingüístico "alto contenido de alcohol"

{

:label "alto contenido de alcohol"

:colname "alcohol"

:f [sigmoid 11.65 9]

}

;estado lingüístico "pH bajo"

{

:label "pH bajo"

:colname "pH"

:f [-sigmoid 3.375 2.93]

}

;estado lingüístico "calidad alta"

{

:label "calidad alta"

:colname "quality"

:f [sigmoid 5.5 4]

}

]

:logic [:GMBC]

:predicate (IMP

(AND "alto contenido de alcohol" "pH bajo") "calidad alta")

}

}

## Parámetros de la tarea de Descubrimiento

A diferencia de la tarea de Evaluación, las tareas de Descubrimiento e Inferencia pueden realizar ajustes en los parámetros de las funciones de pertenencia definidas en los estados lingüísticos. Para ello se pueden especificar estados lingüísticos con funciones de pertenencia de parámetros variables e incluso "estados lingüísticos" sin funciones de pertenencia.

Si se definen estados lingüísticos sin funciones de pertenencia, *universe-cmd* utilizará Funciones de Pertenencia Generalizadas (FPG) con parámetros variables.

Estructura general de una tarea de Descubrimiento

{

:job :discovery

:query

{

; parámetros de la tarea de Descubrimiento

}

}

El valor de la llave *:query* es un mapa que contiene las siguientes llaves:

|  |  |  |  |
| --- | --- | --- | --- |
| Llave | Valor | | Descripción |
| :db-uri | cadena de caracteres | Ruta del set de datos Excel. | |
| :out-file | cadena de caracteres | Ruta del archivo de resultados Excel. | |
| :states | vector de mapas | Estados lingüísticos | |
| :logic | vector | Tipo de lógica difusa a utilizar | |
| :generator | mapa | Generador de predicados difusos | |
| :depth | número entero positivo | Profundidad de los predicados generados a partir del predicado simbólico | |
| :num-pop | número entero positivo | Cantidad de predicados difusos en la población inicial. | |
| :num-iter | número entero positivo | Cantidad de iteraciones (generaciones) de predicados difusos | |
| :num-result | número entero positivo | Cantidad máxima deseada de predicados difusos | |
| :min-truth-value | número real positivo entre 0 y 1 | Valor de verdad mínimo deseado en cada predicado difuso | |
| :mut-percentage | número real positivo entre 0 y 100 | Porcentaje de mutación | |
| :adj-num-pop | número entero positivo | Cantidad de funciones de pertenencia en la población inicial para realizar el ajuste de parámetros | |
|  |  |  | |
| :adj-num-iter | número entero positivo | Cantidad de iteraciones (generaciones) de funciones de pertenencia | |
| :adj-min-truth-value | número real positivo entre 0 y 1 | Valor de verdad mínimo deseado al evaluarse las funciones de pertenencia en el predicado difuso correspondiente | |

### Generador de predicados difusos

El generador de predicados difusos permite especificar la estructura de los predicados difusos que se generarán en la tarea de Descubrimiento mediante el uso de *predicados simbólicos*, vectores de operadores y estados lingüísticos. Este generador será utilizado por el algoritmo genético como plantilla para generar poblaciones de predicados difusos.

### Predicado simbólico

El predicado simbólico es una forma de representación de un predicado difuso en el que se puede utilizar el asterisco "\*" como comodín en sustitución de estados lingüísticos o predicados difusos, de forma similar a las [expresiones regulares](https://es.wikipedia.org/wiki/Expresi%C3%B3n_regular).

El generador de predicados difusos se representa como un mapa con las siguientes llaves:

|  |  |  |
| --- | --- | --- |
| Llave | Valor | Descripción |
| :predicate | lista, vector o "\*" | Predicado simbólico. |
| :operators | lista o vector | Operadores permitidos (puede estar vacío) |
| :variables | lista o vector | Estados lingüísticos permitidos |

La llave *:predicate* establece el predicado simbólico. El algoritmo genético (AG) sustituirá cada asterisco por estados lingüísticos o predicados difusos que incluyan los operadores y estados lingüísticos especificados en las llaves :*operators* y *:variables.* Los predicados difusos que se generan a partir del cada asterisco tendrán el nivel de anidamiento o profundidad especificado en la llave *:depth*.

En dependencia del valor de la llave *:predicate* (predicado simbólico) el generador de predicados difusos puede clasificarse en:

### Generador de predicados difusos con el predicado simbólico "\*"

Este es el predicado simbólico más general que se puede especificar en una tarea de Descubrimiento, en este caso el AG generará poblaciones aleatorias de predicados difusos. En este caso la lista de operadores (llave *:operators*) y estados lingüísticos (llave *:variables*) no pueden estar vacías.

ej.

{:job :discovery

...

:query {:generator {:predicate "\*"

:operators (AND OR IMP EQV NOT)

:variables ("estado1", "estado2","estado3", "estado4")

}

:depth 1

...

En este ejemplo se muestra un fragmento de una tarea de descubrimiento, se han omitido algunas llaves para más claridad.

A continuación, se muestran algunos ejemplos de los predicados difusos que pudieran generarse:

(AND "estado1" "estado3" "estado2" "estado4")

(IMP "estado2" "estado1")

(OR "estado3" "estado4")

(NOT "estado2")

...

Si se hubiera tomado una profundidad de 2 *(:depth 2*), los predicados difusos generados tendrían dos niveles de anidamiento, por ejemplo:

(AND

(NOT "estado1")

(OR "estado3" "estado4")

(EQV "estado1" "estado2"))

(IMP

(AND "estado2" "estado1")(NOT "estado4))

(OR (IMP "estado1" "estado2")(IMP "estado3" "estado4"))

...

### Generador de predicados difusos utilizando asteriscos en el predicado simbólico

Este tipo de generador permite especificar con más detalle la conformación de los predicados difusos que se deseen obtener mediante el uso de uno o varios asteriscos en el predicado simbólico.

ej.1

{

:job :discovery

...

:query {

:generator {

:predicate (IMP "\*" "\*")

:operators (AND OR IMP EQV NOT)

:variables ("estado1", "estado2", "estado3", "estado4")

}

:depth 1

...

}

}

En este caso se desean obtener implicaciones. Cada asterisco será sustituido por predicados difusos de profundidad 1 (*:depth* 1)

A continuación, se muestran algunos ejemplos de los predicados difusos que pudieran generarse:

(IMP

(OR "estado1" "estado3")

(AND "estado3" "estado1" "estado2"))

(IMP

(NOT "estado2")

(IMP "estado3" "estado2"))

...

ej.2

{

:job :discovery

...

:query {

:generator {

:predicate (IMP (AND "\*" "\*") "estado4")

:operators ()

:variables ("estado1", "estado2",

"estado3")

}

:depth 1

...

}

}

En este ejemplo se desean obtener predicados difusos donde la conjunción de dos estados lingüísticos impliquen el *"estado4".*

Nótese que en este caso el valor de la llave *:operators* está vacía, por lo que los asteriscos sólo podrán ser reemplazados por los estados lingüísticos definidos en la llave *:variables*. En la llave *:variables* se ha omitido el *"estado4"* para evitar que el AG la utilice en la premisa del operador implicación (IMP), que en este caso es el operador conjunción (AND). Además el valor de la llave *:depth* será ignorado ya que la lista de operadores está vacía.

A continuación, se muestran algunos ejemplos de los predicados difusos que pudieran generarse:

(IMP

(AND "estado1" "estado3")

"estado4")

(IMP

(AND "estado2" "estado1" "estado3")

"estado4")

...

### Generador de predicados difusos sin utilizar asteriscos en el predicado simbólico

Se puede definir predicados simbólicos sin utilizar asteriscos, es decir, el predicado simbólico será un predicado difuso.

En este caso la tarea de Descubrimiento se comportará de forma similar a la tarea de Evaluación, con la diferencia que ajustará los parámetros de los estados lingüísticos con funciones de pertenencia variables. Al no haber asteriscos en predicado simbólico, el AG ignorará los valores de las llaves *:operators*, *:variables* y *:depth*.

ej.

{

:job :discovery

...

:query {

:generator {

:predicate (IMP "estado1" "estado4")

:operators ()

:variables ()

}

:depth 1

...

}

}

### Ajuste de los parámetros de las funciones de pertenencia variables.

Cuando se realiza la tarea de Descubrimiento utilizando estados lingüísticos con funciones de pertenencia de parámetros variables, se realizará el ajuste de dichos parámetros, este ajuste se realizará mediante un AG que para cada predicado difuso generado creará una población de *:adj-num-pop* funciones de pertenencia variando sus parámetros con el fin de maximizar el valor de verdad del predicado a un valor mínimo *:adj-min-truth-value*, el AG realizará *:adj-num-iter* iteraciones.

El ajuste se realizará

Si las funciones de pertenencias tienen parámetros fijos, no se podrá realizar el ajuste por lo que se ignorarán los valores de las llaves *:adj-num-pop, :adj-min-truth-value* y *:adj-num-iter*

## Parámetros de la tarea de Inferencia

La tarea de inferencia realiza una tarea de Descubrimiento para obtener predicados difusos que relacionan las variables de condición con las variables de decisión, una vez realizado el Descubrimiento se utilizarán los predicados difusos obtenidos para inferir los valores de las variables de decisión utilizando un algoritmo genético.

Estructura general de una tarea de Inferencia

{

:job :inference

:query

{

; parámetros de la tarea de Inferencia

}

}

El valor de la llave *:query* es un mapa que contiene las siguientes llaves:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Llave | | Valor | | Descripción |
| :db-uri | | cadena de caracteres | Ruta del set de datos Excel donde se realizará el Descubrimiento de los predicados difusos que relacionan las variables de condición con las de decisión | |
| :condition-db | | cadena de caracteres | Ruta del set de datos que contienen las variables de condición que se utilizarán para inferir las variables de decisión | |
| :out-file | | cadena de caracteres | Ruta del archivo de resultados Excel. | |
| :states |  | mapa | Estados lingüísticos | |
|  | :condition | vector de mapas | Estados lingüísticos de las variables de condición | |
| :decision | vector de mapas | Estados lingüísticos de las variables de decisión | |
| :discovery |  | mapa | Parámetros para el Descubrimiento de los predicados difusos que relacionan las variables de condición con las de decisión | |
|  | :logic | vector | Tipo de lógica difusa a utilizar | |
| :depth | número entero positivo | Profundidad de los predicados generados | |
| :num-pop | número entero positivo | Cantidad de predicados difusos en la población inicial. | |
| :num-iter | número entero positivo | Cantidad de iteraciones (generaciones) de predicados difusos | |
| :num-result | número entero positivo | Cantidad máxima deseada de predicados difusos | |
| :min-truth-value | número real positivo entre 0 y 1 | Valor de verdad mínimo deseado en cada predicado difuso | |
| :mut-percentage | número real positivo entre 0 y 100 | Porcentaje de mutación | |
| :adj-num-pop | número entero positivo | Cantidad de funciones de pertenencia en la población inicial para realizar el ajuste de parámetros | |
| :adj-num-iter | número entero positivo | Cantidad de iteraciones (generaciones) de funciones de pertenencia | |
| :adj-min-truth-value | número real positivo entre 0 y 1 | Valor de verdad mínimo deseado al evaluarse las funciones de pertenencia en el predicado difuso correspondiente | |
| :method | | "IMP" o "EQV" | Método a utilizar para realizar la inferencia de variables de decisión | |
| :num-pop | | número entero positivo | Población inicial de valores de variables de decisión. | |
| :num-iter | | número entero positivo | Cantidad de iteraciones (generaciones) de valores de variables de decisión. | |
| :num-result | | número entero positivo | Cantidad máxima deseada de valores de variables de decisión. | |
| :min-truth-value | | número real positivo entre 0 y 1 | Valor de verdad mínimo deseado para la obtención de valores de variables de decisión. | |
| :mut-percentage | | número real positivo entre 0 y 100 | Porcentaje de mutación | |

La tarea de Inferencia utilizará internamente el Descubrimiento de predicados difusos con el predicado simbólico "\*" y los operadores *AND, OR, NOT, IMP y EQV* garantizando que en los predicados difusos obtenidos estén presentes todas las variables de decisión y al menos una variable de condición.

### Método a utilizar para realizar la inferencia de variables de decisión (llave :method)

La llave :method puede tomar valor "EQV" o "IMP" para realizar la inferencia.

### Método "EQV"

Si se utiliza el método "EQV" se tomará cada población de valores de variables de decisión y las variables de condición y se evaluarán en cada predicado difuso de forma tal que se cumpla que el valor de verdad obtenido sea equivalente al del predicado difuso. El AG maximizará la conjunción de todos los predicados difusos.

### Método "IMP"

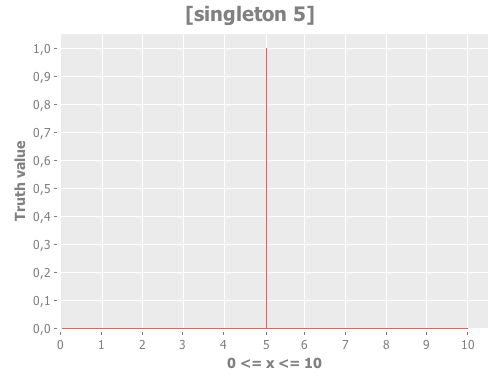
Si se utiliza el método "IMP" se tomará cada población de valores de variables de decisión y las variables de condición y se evaluarán en cada predicado difuso de forma tal que se cumpla que el valor de verdad obtenido Implique la equivalencia del predicado difuso y el valor obtenido. El AG maximizará la conjunción de todos los predicados difusos.

# Anexo 1. Funciones de Pertenencia.

## Funciones de pertenencia discretas

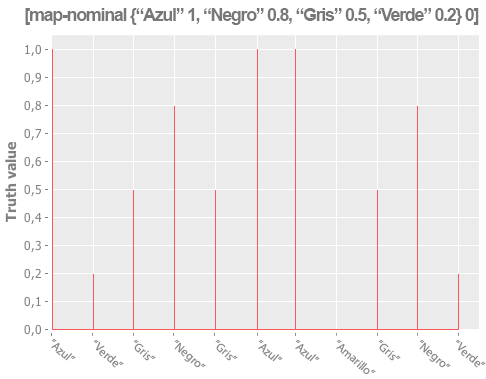
Nombre: **singleton**

Parámetros: **a**



Nombre: **map-nominal**

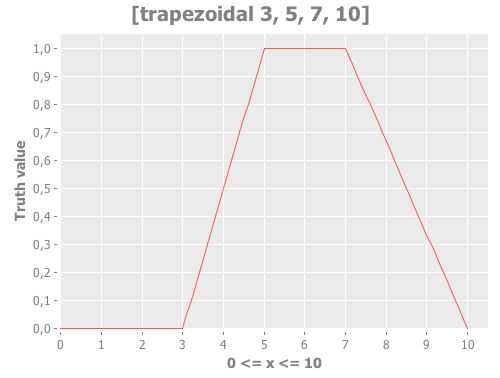
Parámetros: **values-map, not-found-value**



## Funciones de pertenencia continuas

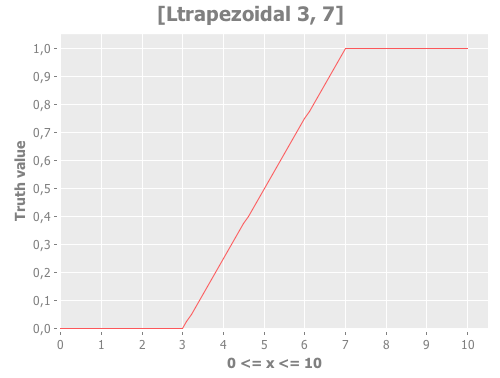
Nombre: **trapezoidal**

Parámetros: **a, b, c, d**



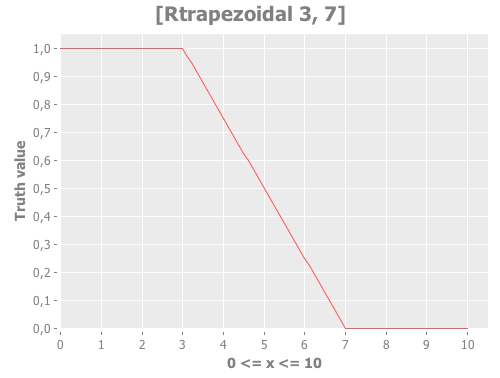
Nombre: **Ltrapezoidal**

Parámetros: **a, b**



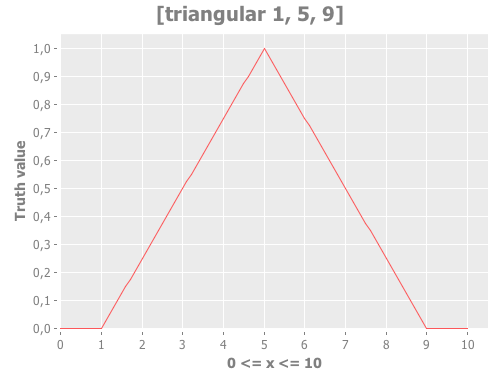
Nombre: **Rtrapezoidal** *Media trapezoidal derecha*

Parámetros: **a, b**



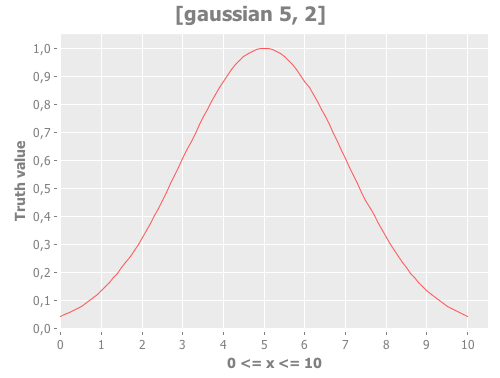
Nombre: **triangular**

Parámetros: **a, b, c**



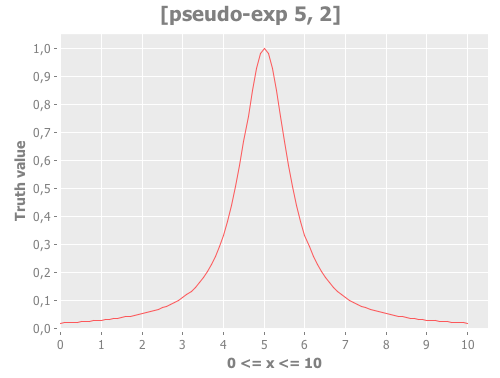
Nombre: **gaussian**

Parámetros: **center, deviation**



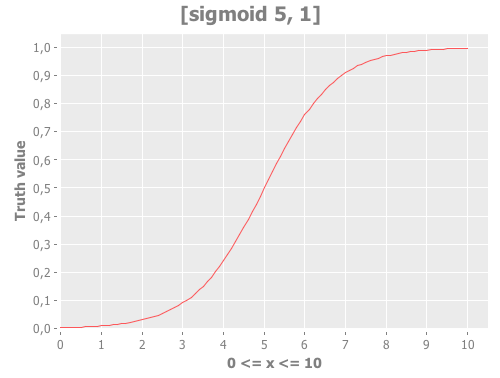
Nombre: **pseudo-exp**

Parámetros: **center, deviation**



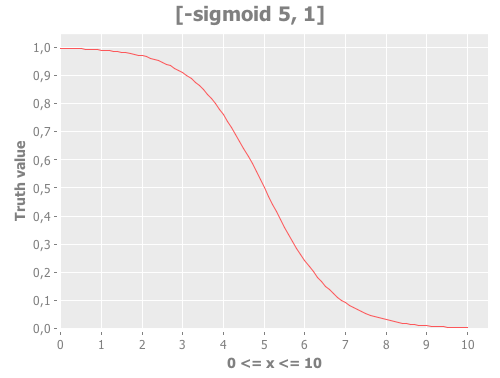
Nombre: **sigmoid**

Parámetros: **center, beta**



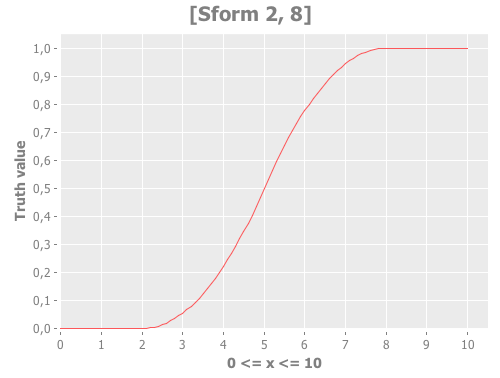
Nombre: **-sigmoid**

Parámetros: **center, beta**



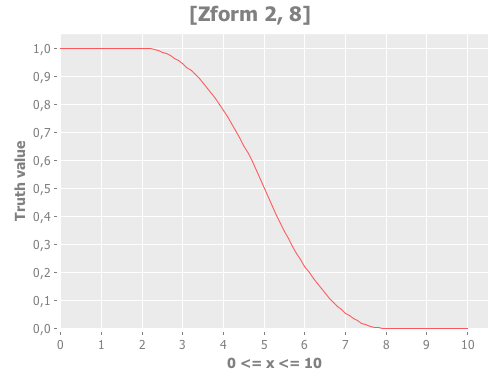
Nombre: **Sform**

Parámetros: **a, b**



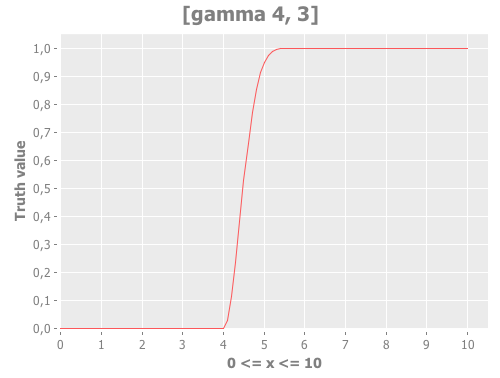
Nombre: **Zform**

Parámetros: **a, b**



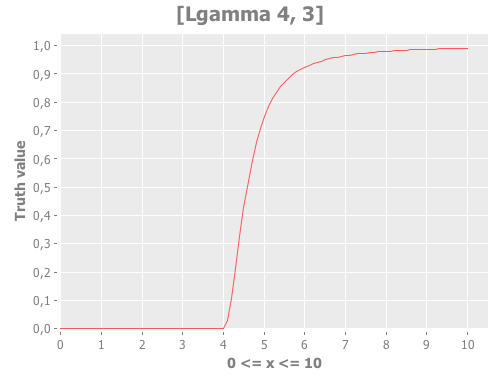
Nombre: **gamma**

Parámetros: **a, b**



Nombre: L**gamma**

Parámetros: **a, b**



PENDIENTES

-Agregar relación de orden entre los valores de los parámetros

-Agregar la ecuación que define a cada función de pertenencia