Revista digital
Matemática, Educación e Internet
(http://www.tec-digital.itcr.ac.cr/revistamatematica/).
Vol 15, No 1. Agosto – Febrero 2015.
Artículo de sección
ISSN 1659 -0643

Paquete `VilGebra`: recurso didáctico a través del uso del software *Mathematica* en el campo del álgebra lineal

Enrique Vílchez Q.
enrique.vilchez.quesada@una.cr
Escuela de Informática
Universidad Nacional de Costa Rica

Recibido: 22 Octubre, 2013 Aceptado: 2 Mayo, 2014

Resumen. **`VilGebra`** es un paquete elaborado por el autor de esta propuesta, que añade al software comercial *Mathematica 9*, ochenta y seis comandos para desarrollar distintos procedimientos vinculados con un curso introductorio de álgebra lineal para ingeniería. En general, las instrucciones integradas en **`VilGebra`** recorren ejes temáticos relacionados con: matrices y determinantes, vectores en $\mathbb{R}^n$, rectas y planos, espacios vectoriales reales, proyecciones ortogonales, transformaciones lineales, valores y vectores propios, diagonalización de matrices y operadores lineales y, programación lineal. El trabajo es producto de la experiencia docente del autor, impartiendo cursos asistidos por computadora en esta importante área de conocimiento.

Palabras clave: Álgebra, lineal, software, *Mathematica*, enseñanza, aprendizaje.

Abstract. **`VilGebra`** is a package developed by the author of this proposal, which adds to the commercial software *Mathematica* 9, eighty-six different commands to develop procedures associated with an introductory course in linear algebra to engineering. In general, integrated instructions roam **`VilGebra`** related themes: matrices and determinants, vectors in $\mathbb{R}^n$, lines and planes, real vector spaces, orthogonal, linear transformations, eigenvalues and eigenvectors, diagonalization of matrices and linear operators and linear programming. The work is the product of the author teaching experience, computer-assisted teaching courses in this important area of knowledge.

KeyWords: algebra, linear, software, *Mathematica*, teaching, learning.

1.1 Introducción

La matemática asistida por computadora es un área de conocimiento que brinda importantes opciones metodológicas frente a los retos que impone una enseñanza mediada y un aprendizaje de naturaleza heurística. Ante estos desafíos, la educación matemática tradicional a nivel universitario, debe circunscribir sus esfuerzos por rescatar el valor lógico, instrumental y práctico de esta disciplina en la vida cotidiana de los estudiantes y en la resolución de problemas relacionados con el contexto. Torres a este respecto señala: "*una de las principales dificultades ... tiene que ver con la producción de situaciones problemáticas relacionadas con el mundo real*" (2006, p. 754). En este sentido, el uso de software en las prácticas de aprendizaje de los alumnos, puede aproximar los contenidos matemáticos formales de manera natural y contextualizada, a los campos de conocimiento particulares en cursos vinculados con ingeniería. Esta idea se fortalece pues a través del uso de software es posible resolver problemas próximos a la realidad del estudiantado, donde usualmente se involucra una cantidad de variables muy significativa, difícil de manejar utilizando únicamente procedimientos de carácter manual.

El álgebra lineal es una de las áreas más importantes de desarrollo en la formación matemática de cualquier futuro ingeniero. Vílchez y Monge (2001) así lo destacan justificando este hecho por sus innumerables aplicaciones en otros campos disciplinarios. Bajo esta perspectiva, su enseñanza debería ir más allá de la simple gesta de aprender algoritmos y su aplicación práctica.

La experiencia docente desarrollada durante varios años de experimentación de aula, ha llevado al autor del presente trabajo, a la creación de un paquete denominado "**VilGebra**", vinculado con el software *Mathematica*, que reúne las condiciones necesarias y suficientes para complementar una metodología tradicional propia de un curso universitario de álgebra lineal, con otra asistida por computadora. La fórmula perfecta reside en buscar el adecuado y difícil equilibrio entre la formalidad y rigurosidad de los contenidos, con la resolución de problemas mediados de manera transparente por un ordenador.

La transparencia en el uso de medios tecnológicos para la producción cognitiva, no es una tarea simple, requiere muchas veces facilitar el uso de las herramientas tecnológicas que entran en el ambicioso juego del aprendizaje. **VilGebra** apunta a facilitar la exploración de las extraordinarias capacidades del software *Mathematica* como una aplicación de carácter científico, sin implicar al alumnado, en el entrañable rol de la programación y sus complejas habilidades, poco probables de adquirir en el corto período de un semestre.

A continuación se presentan las ochenta y seis funciones integradas en **VilGebra**, recurriendo a ejemplos clásicos dentro de los distintos capítulos que usualmente constituyen una asignatura de álgebra lineal para ingeniería.

1.2 Instalación del paquete VilGebra

VilGebra se ha creado para versiones de *Mathematica* superiores o iguales a la 8.0. Para instalar dicho paquete se accede al menú en el software *Mathematica*: **File/Install**. Lo anterior, despliega la siguiente ventana:

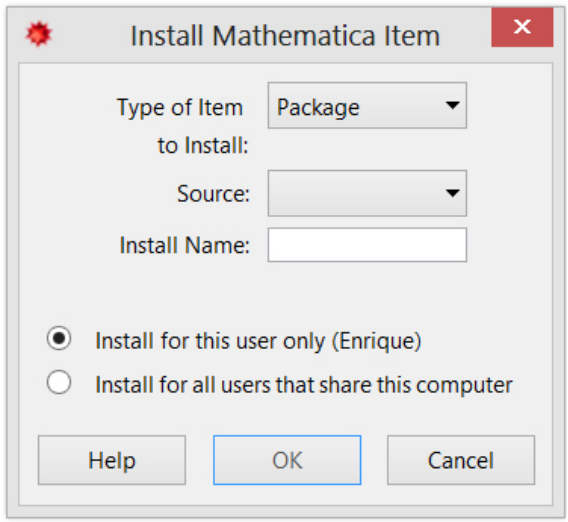

Figura 1: Instalación de **VilGebra**

La opción **Source** en la figura 1 permite direccionar el archivo del paquete que debe estar descargado localmente en la computadora del usuario (en este sitio Web se encuentra disponible la última versión de **VilGebra**). Se recomienda en **Install Name** dejar por defecto el nombre **VilGebra**.

Si se desea levantar el paquete en un **notebook** de *Mathematica* se utiliza la instrucción: << **VilGebra'**. Esto le permite al estudiante tener acceso a todas las funciones incluidas en la biblioteca.

Los ejemplos y ejercicios de este documento se encuentran resueltos con detalle en el archivo: "*Ejemplos resueltos*" disponible en la dirección URL:

`http://www.tec-digital.itcr.ac.cr/revistamatematica/Secciones/Didactica_y_Software/RevistaDigital_Vilchez_V15_n1_2014/VilGebra.rar`

1.3 Matrices y determinantes con VilGebra

El software *Mathematica* no cuenta con funciones explícitas para realizar ciertas operaciones con matrices. Por ejemplo, las operaciones elementales de fila, procedimientos paso a paso para reducir una matriz a su forma escalonada, encontrar la inversa de una matriz invertible y hallar la solución de un sistema de ecuaciones lineales. En esta sección, se muestran los comandos integrados en el paquete *VilGebra* que intentan llenar estos vacíos y añadir al programa otras herramientas clave, para el cálculo de de un determinante, la inversa por cofactores, el estudio de vectores li y ld y el tema de combinaciones lineales.

1. **`IntercambiaFila`**: intercambia dos filas en una matriz. Sintaxis: **`IntercambiaFila[A, fi, fj]`**, siendo **`fi`** y **`fj`** número naturales que representan las filas a intercambiar.

2. **`MultiFila`**: multiplica un escalar a una fila de una matriz. Sintaxis: **`MultiFila[A, esc, fi]`**, en este caso, se multiplica el escalar **`esc`** a la fila **`fi`**.

3. **`SumaFila`**: suma el múltiplo escalar de una fila en una matriz a otra fila dada. Sintaxis: **`SumaFila[A, esc, fj, fi]`**, esta instrucción realiza la operación **`esc fj + fi`**, es decir, suma a la fila **`fi`** el mútiplo escalar **`esc`** de la fila **`fj`**.

Ejemplo 1.1

Encuentre la matriz reducida por filas de:

$$A = \begin{pmatrix} 1 & 2 & 3 \\ 5 & 6 & 7 \\ 9 & 8 & 5 \end{pmatrix}$$

Solución: En este caso se utilizan los comandos expuestos en 2 y 3 como se especifica a continuación:

- **`SumaFila[A, -5, 1, 2]`**
- **`SumaFila[%, -9, 1, 3]`**
- **`MultiFila[%, -1/4, 2]`**
- **`SumaFila[%, -2, 2, 1]`**
- **`SumaFila[%, 10, 2, 3]`**
- **`MultiFila[%, -1/2, 3]`**
- **`SumaFila[%, 1, 3, 1]`**
- **`SumaFila[%, -2, 3, 2]`**

Lo anterior produce como salida la matriz identidad. Se aclara al lector que el símbolo % en *Mathematica*, recoge la última salida producida por el software.

4. **`RowReduceComplete`**: función que determina paso a paso la matriz escalonada de otra, recibida como parámetro. Sintaxis: **`RowReduceComplete[A]`**, con **`A`** la matriz que se desea reducir.

Ejemplo 1.2

Encuentre la matriz reducida por filas de:

$$A = \begin{pmatrix} 1 & 1 & -1 & 1 & 1 \\ -2 & 1 & 2 & 1 & 1 \\ 1 & 2 & -2 & -3 & -2 \end{pmatrix}$$

Solución: El comando **RowReduceComplete** determina el proceso de reducción paso a paso. En *Mathematica*:

In[ ]:= **RowReduceComplete** $\left[\begin{pmatrix} 1 & 1 & -1 & 1 & 1 \\ -2 & 1 & 2 & 1 & 1 \\ 1 & 2 & -2 & -3 & -2 \end{pmatrix}\right]$

$\vdots$

Out[ ]= $\begin{pmatrix} 1 & 0 & -1 & 0 & 0 \\ 0 & 1 & 0 & 1 & 1 \\ 0 & 0 & -1 & -5 & -4 \end{pmatrix} \begin{pmatrix} 1 & 0 & -1 & 0 & 0 \\ 0 & 1 & 0 & 1 & 1 \\ 0 & 0 & 1 & 5 & 4 \end{pmatrix} \begin{pmatrix} 1 & 0 & 0 & 5 & 4 \\ 0 & 1 & 0 & 1 & 1 \\ 0 & 0 & 1 & 5 & 4 \end{pmatrix}$

- Es importante aclarar al lector que *Mathematica* cuenta con el comando **RowReduce**, sin embargo, esta instrucción determina la matriz reducida por filas sin mostrar el procedimiento empleado.

Ejercicio

Utilizando el comando **RowReduceComplete** encuentre la matriz escalonada equivalente por filas a cada una de las matrices dadas. Compare el resultado con el devuelto por **RowReduce**.

(a) $A = \begin{pmatrix} 1 & 1 & -1 & \sqrt{2} \\ 1 & 2 & -1 & -2 \\ 5 & 4 & -\frac{2}{3} & 0 \end{pmatrix}$

(b) $B = \begin{pmatrix} 1 & 1 & 1 \\ 2 & -3 & 1 \\ 1 & -5 & \frac{2}{3} \\ 1 & 1 & 1 \end{pmatrix}$

5. **InversaSP**: encuentra paso a paso por el método de *Jordan-Gauss*, la matriz inversa de otra que no contiene ningún parámetro. Sintaxis: **InversaSP[A]**, siendo **A** la matriz correspondiente.

Ejemplo 1.3

Determine A^{-1} por el método de *Jordan-Gauss*, siendo:

$$A = \begin{pmatrix} 0 & 2 & 3 \\ 0 & 6 & 7 \\ 9 & 8 & 5 \end{pmatrix}$$

Solución: En el software:

In[ ]:=

$$A = \begin{pmatrix} 0 & 2 & 3 \\ 0 & 6 & 7 \\ 9 & 8 & 5 \end{pmatrix};$$

InversaSP$[A]$

Out[ ]=

$$\vdots$$

$$\begin{pmatrix} 1 & 0 & 0 & \frac{13}{18} & -\frac{7}{18} & \frac{1}{9} \\ 0 & 1 & \frac{3}{2} & \frac{1}{2} & 0 & 0 \\ 0 & 0 & 1 & \frac{3}{2} & -\frac{1}{2} & 0 \end{pmatrix} \begin{pmatrix} 1 & 0 & 0 & \frac{13}{18} & -\frac{7}{18} & \frac{1}{9} \\ 0 & 1 & 0 & -\frac{7}{4} & \frac{3}{4} & 0 \\ 0 & 0 & 1 & \frac{3}{2} & -\frac{1}{2} & 0 \end{pmatrix}$$

La matriz inversa corresponde a: $\begin{pmatrix} \frac{13}{18} & -\frac{7}{18} & \frac{1}{9} \\ -\frac{7}{4} & \frac{3}{4} & 0 \\ \frac{3}{2} & -\frac{1}{2} & 0 \end{pmatrix}$

- Las siglas "**SP**" de la instrucción **InversaSP** significan "sin parámetros", es decir, todas las entradas de la matriz que se toma como argumento son números reales o complejos.

6. **InversaCP**: encuentra paso a paso la matriz inversa de otra que contiene parámetros. Sintaxis: **InversaCP[A]**, siendo **A** la matriz con parámetros.

Ejemplo 1.4

Halle A^{-1} donde:

$$A = \begin{pmatrix} y & 2 & 3 \\ x & 6 & 7 \\ 9 & x+1 & 5 \end{pmatrix}$$

Solución: Al emplear la función **InversaCP** se obtiene:

In[ ]:=

$$A = \begin{pmatrix} y & 2 & 3 \\ x & 6 & 7 \\ 9 & x+1 & 5 \end{pmatrix};$$

InversaCP$[A]$

Out[ ]=

$$\begin{pmatrix} 1 & 0 & 0 & -\frac{7x-23}{3x^2-7yx-7x+23y-36} & \frac{3x-7}{3x^2-7yx-7x+23y-36} & -\frac{4}{3x^2-7yx-7x+23y-36} \\ 0 & 1 & 0 & -\frac{5x-63}{3x^2-7yx-7x+23y-36} & -\frac{5y-27}{-3x^2+7yx+7x-23y+36} & \frac{3x-7y}{3x^2-7yx-7x+23y-36} \\ 0 & 0 & 1 & \frac{x^2+x-54}{3x^2-7yx-7x+23y-36} & -\frac{xy+y-18}{3x^2-7yx-7x+23y-36} & -\frac{2(x-3y)}{3x^2-7yx-7x+23y-36} \end{pmatrix}$$

La matriz inversa es: $\begin{pmatrix} -\frac{7x-23}{3x^2-7yx-7x+23y-36} & \frac{3x-7}{3x^2-7yx-7x+23y-36} & -\frac{4}{3x^2-7yx-7x+23y-36} \\ -\frac{5x-63}{3x^2-7yx-7x+23y-36} & -\frac{5y-27}{-3x^2+7yx+7x-23y+36} & \frac{3x-7y}{3x^2-7yx-7x+23y-36} \\ \frac{x^2+x-54}{3x^2-7yx-7x+23y-36} & -\frac{xy+y-18}{3x^2-7yx-7x+23y-36} & -\frac{2(x-3y)}{3x^2-7yx-7x+23y-36} \end{pmatrix}$

- Como el lector habrá previsto el "**CP**" del nombre de este comando significa "con parámetros", es decir, la matriz para la cual se desea calcular su inversa tiene algunas entradas que corresponden a variables.

7. **Determinante**: función que calcula el determinante de una matriz cuadrada desarrollando por cofactores de acuerdo con los elementos de la primera fila. Sintaxis: **Determinante[A]**, donde **A** es la matriz cuadrada respectiva.

Ejemplo 1.5

Encuentre el determinante de la matriz:

$$A = \begin{pmatrix} 1 & 2 & 3 & -8 \\ 4 & 5 & 3 & -1 \\ 1 & 2 & -8 & 9 \\ 0 & 1 & -7 & 9 \end{pmatrix}$$

Solución: Al recurrir al comando **Determinante**:

In[ ]:= Determinante $\left[\begin{pmatrix} 1 & 2 & 3 & -8 \\ 4 & 5 & 3 & -1 \\ 1 & 2 & -8 & 9 \\ 0 & 1 & -7 & 9 \end{pmatrix}\right]$

Out[ ]=

$$\begin{pmatrix} 5 & 3 & -1 \\ 2 & -8 & 9 \\ 1 & -7 & 9 \end{pmatrix}\begin{pmatrix} 4 & 3 & -1 \\ 1 & -8 & 9 \\ 0 & -7 & 9 \end{pmatrix}\begin{pmatrix} 4 & 5 & -1 \\ 1 & 2 & 9 \\ 0 & 1 & 9 \end{pmatrix}\begin{pmatrix} 4 & 5 & 3 \\ 1 & 2 & -8 \\ 0 & 1 & -7 \end{pmatrix}$$

128

Las cuatro matrices mostradas en la salida, corresponden a las generadas por los cofactores del cálculo del determinante, de acuerdo con los elementos de la primera fila. Sería interesante como una tarea futura para el lector, modificar el código de esta función para crear otra, donde el usuario seleccione la fila o columna con respecto a la cual se desea calcular el determinante.

8. **InversaAdjunta**: calcula la inversa por el método de la adjunta donde cada uno de los cofactores son hallados paso a paso. Sintaxis: **InversaAdjunta[A]**.

Ejemplo 1.6

Determine la inversa de la matriz dada, por el método de la adjunta.

$$A = \begin{pmatrix} 1 & 2 & 3 \\ 5 & 6 & 7 \\ 9 & 8 & 5 \end{pmatrix}$$

Solución: En *Mathematica*:

In[ ]:=

$$A = \begin{pmatrix} 1 & 2 & 3 \\ 5 & 6 & 7 \\ 9 & 8 & 5 \end{pmatrix};$$

InversaAdjunta$[A]$

Out[ ]=

La matriz adjunta es: $\begin{pmatrix} -26 & 14 & -4 \\ 38 & -22 & 8 \\ -14 & 10 & -4 \end{pmatrix}$

La matriz inversa corresponde a: $\begin{pmatrix} -\frac{13}{4} & \frac{7}{4} & -\frac{1}{2} \\ \frac{19}{4} & -\frac{11}{4} & 1 \\ -\frac{7}{4} & \frac{5}{4} & -\frac{1}{2} \end{pmatrix}$

Recurriendo al paquete **Combinatoria** que trae por defecto *Mathematica*, es posible crear otra función que calcula la inversa de una matriz utilizando el método de la adjunta. A continuación se muestra el código correspondiente:

```
Quiet[<< Combinatorica‘]
OInversaAdjunta[T_List] :=
 Module[{n = Dimensions[A][[1]]},
  NADJ = ConstantArray[0, Dimensions[T]];
  For[i = 1, i <= n,
   For[j = 1, j <= n, NADJ[[i, j]] = Cofactor[T, {i, j}]; j++]; i++;
   If [i == n + 1,
    Print["La matriz adjunta es: ", MatrixForm[Transpose[NADJ]]]]];
  If[Det[T] == 0, Print["La matriz es singular"],
   Print["La matriz inversa corresponde a: ",
    MatrixForm[1/Det[T] Transpose[NADJ]]]]]
```

El método anterior emplea el comando **Cofactor** para hallar cada uno de los cofactores requeridos en el cálculo de la matriz inversa. Esta instrucción forma parte del paquete Combinatorica.

9. **LinearSolveCompleteSP**: resuelve paso a paso un sistema de ecuaciones lineales sin parámetros. Sintaxis: **LinearSolveCompleteSP[A, B]**, siendo **A** la matriz de coeficientes numéricos del sistema y **B** la matriz de términos constantes.

Ejemplo 1.7

Determine la solución del siguiente sistema de ecuaciones lineales:

$$\begin{cases} x+y-z+w+v=1 \\ -2x+y+2z+w+v=2 \\ x+2y-2z-3w-2v=-1 \end{cases}$$

Solución: Mediante el uso de LinearSolveCompleteSP se obtiene:

In[ ]:= LinearSolveCompleteSP
$$\left[\begin{pmatrix} 1 & 1 & -1 & 1 & 1 \\ -2 & 1 & 2 & 1 & 1 \\ 1 & 2 & -2 & -3 & -2 \end{pmatrix}, \begin{pmatrix} 1 \\ 2 \\ -1 \end{pmatrix}\right]$$

Out[ ]=

$$\vdots$$

$$\begin{pmatrix} 1 & 0 & -1 & 0 & 0 & -\frac{1}{3} \\ 0 & 1 & 0 & 1 & 1 & \frac{4}{3} \\ 0 & 0 & -1 & -5 & -4 & -\frac{10}{3} \end{pmatrix} \begin{pmatrix} 1 & 0 & -1 & 0 & 0 & -\frac{1}{3} \\ 0 & 1 & 0 & 1 & 1 & \frac{4}{3} \\ 0 & 0 & 1 & 5 & 4 & \frac{10}{3} \end{pmatrix} \begin{pmatrix} 1 & 0 & 0 & 5 & 4 & 3 \\ 0 & 1 & 0 & 1 & 1 & \frac{4}{3} \\ 0 & 0 & 1 & 5 & 4 & \frac{10}{3} \end{pmatrix}$$

Lo interesante del comando radica en mostrar paso a paso la aplicación del método de *Jordan-Gauss* para encontrar la solución del sistema de ecuaciones lineales. De la última matriz se concluye: $S = \left\{ \left(3-5w-4v, \frac{4}{3}-w-v, \frac{10}{3}-5w-4v, w, v\right) \mid w,v \in \mathbb{R} \right\}$.

Ejercicio

Encuentre el conjunto solución de los siguientes sistemas de ecuaciones lineales.

(a) $\begin{cases} x+y-z+\sqrt{2}w=1 \\ x+2y-z-2w=2 \\ 5x+4y-\frac{2}{3}z=5 \end{cases}$

(b) $\begin{cases} x+y+z=1 \\ 2x-3y+z=-2 \\ x-5y+\frac{2}{3}z=-3 \\ x+y+6\sqrt{3}z=\sqrt{5} \end{cases}$

10. **`LinearSolveCompleteCP`**: resuelve paso a paso un sistema de ecuaciones lineales con parámetros. Sintaxis: **`LinearSolveCompleteCP[A, B]`**, con **`A`** y **`B`** las mismas matrices del comando anterior.

Ejemplo 1.8

Halle los valores de k para los cuales el siguiente sistema de ecuaciones lineales tiene: solución única, infinidad de soluciones o solución vacía.

$$\begin{cases} kx-y+kz=k \\ x+y+z=2k \\ -x+ky+z=2 \end{cases}$$

Solución: En *Mathematica*:

`In[ ]:= LinearSolveCompleteCP`
$$\left[\begin{pmatrix} k & -1 & k \\ 1 & 1 & 1 \\ -1 & k & 1 \end{pmatrix}, \begin{pmatrix} k \\ 2k \\ 2 \end{pmatrix}\right]$$

`Out[ ]=`

$$\vdots$$

$$\begin{pmatrix} 1 & 0 & 0 & \frac{(k-1)\left(2k^2+k+2\right)}{2(k+1)} \\ 0 & 1 & 0 & \frac{k(2k-1)}{k+1} \\ 0 & 0 & 2 & -(k-2)(2k+1) \end{pmatrix} \begin{pmatrix} 1 & 0 & 0 & \frac{(k-1)\left(2k^2+k+2\right)}{2(k+1)} \\ 0 & 1 & 0 & \frac{k(2k-1)}{k+1} \\ 0 & 0 & 2 & -(k-2)(2k+1) \end{pmatrix}$$

$$\begin{pmatrix} 1 & 0 & 0 & \frac{(k-1)\left(2k^2+k+2\right)}{2(k+1)} \\ 0 & 1 & 0 & \frac{k(2k-1)}{k+1} \\ 0 & 0 & 1 & -\frac{1}{2}(k-2)(2k+1) \end{pmatrix}$$

Al observar el proceso de reducción paso a paso, es posible identificar las restricciones $k \neq 0, -1$ para que el sistema tenga solución única. Veamos qué ocurre si $k=0$ y $k=-1$:

In[ ]:=

$k = 0;$

RowReduce $\left[\begin{pmatrix} k & -1 & k & k \\ 1 & 1 & 1 & 2k \\ -1 & k & 1 & 2 \end{pmatrix}\right]$

Out[ ]= $\begin{pmatrix} 1 & 0 & 0 & -1 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 \end{pmatrix}$

En este caso, el sistema continúa teniendo solución única. Además:

In[ ]:=

$k = -1;$

RowReduce $\left[\begin{pmatrix} k & -1 & k & k \\ 1 & 1 & 1 & 2k \\ -1 & k & 1 & 2 \end{pmatrix}\right]$

Out[ ]= $\begin{pmatrix} 1 & 0 & 0 & -1 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 \end{pmatrix}$

Para $k = -1$ la solución es el conjunto vacío. Finalmente, se concluye que el sistema tiene solución única si $k \in \mathbb{R} - \{-1\}$, nunca tendrá infinidad de soluciones y si $k = -1$ tendrá solución vacía.

Ejercicio

Encuentre los valores de k para los cuales el siguiente sistema de ecuaciones lineales tiene: solución única, infinidad de soluciones o solución vacía.

$$\begin{cases} x + ky + 3z = 3 \\ 2kx + ky + 2kz = -1 \\ kx + ky + kz = -\sqrt{3} \end{cases}$$

11. **Cramer**: resuelve un sistema de ecuaciones lineales cuadrado a partir de la regla de *Cramer*. Sintaxis: **Cramer[A, B]**, con **A** la matriz de coeficientes numéricos y **B**un vector fila que representa la matriz de términos constantes del sistema.

Ejemplo 1.9

Recurriendo a la regla de *Cramer* resuelva:

$$\begin{cases} 3x+2y+z=1 \\ x-2y+z=0 \\ x+y-3z=2 \end{cases}$$

Solución: El comando **`Cramer`** retorna:

`In[ ]:= Cramer` $\left[\begin{pmatrix} 3 & 2 & 1 \\ 1 & -2 & 1 \\ 1 & 1 & -3 \end{pmatrix}, \{1,0,2\}\right]$

`Out[ ]=` $x1=\frac{1}{2}, x2=0, x3=-\frac{1}{2}$

12. **`CombiLinealQ`**: determina si un vector es combinación lineal de un conjunto de vectores dados, retornando True o False en cada caso. Sintaxis: **`CombiLinealQ[Vector, Conjunto]`**, siendo **`Vector`** el vector que se desea determinar si es o no, combinación lineal del conjunto de vectores añadidos en **`Conjunto`**.

Ejemplo 1.10

Establezca si el vector $v=(-10,4)$ es combinación lineal de los vectores del conjunto $\{(1,4),(5,2)\}$.

Solución: En *Mathematica*:

`In[ ]:= CombiLinealQ` $\left[\{-10,4\}, \begin{pmatrix} 1 & 4 \\ 5 & 2 \end{pmatrix}\right]$

`Out[ ]= True`

Se infiere del **`True`** que la combinación lineal de v con respecto a los elementos de $\{(1,4),(5,2)\}$, es verdadera. El lector debe recordar que en *Mathematica* cualquier comando terminado en "**`Q`**" es booleano, es decir, retorna **`True`** o **`False`**. **`VilGebra`** ha respetado también esta convención.

Ejercicio

Señale si el vector v dado es combinación lineal de la familia de vectores adjunta.

(a) $v=(11,2,-7)$ con respecto a $\{(1,2,3),(5,6,1)\}$

(b) $v = (7,10,7)$ con respecto a $\{(1,2,3),(5,6,1)\}$

13. **LiLd**: establece si un conjunto de vectores son linealmente independientes o linealmente dependientes. Sintaxis: **LiLd[A]**, los vectores se escriben como las filas de la matriz **A**.

Ejemplo 1.11

Detemine si el conjunto de vectores $\left\{(1,2,3),(-1,0,4),(1,-1,1),\left(\sqrt{2},1,\sqrt{3}\right)\right\}$ es linealmente independiente (*li*) o linealmente dependiente (*ld*).

Solución: Mediante el uso de la instrucción **LiLd** se tiene:

In[ ]:= LiLd
$$\left[\begin{pmatrix} 1 & 2 & 3 \\ -1 & 0 & 4 \\ 1 & -1 & 1 \\ \sqrt{2} & 1 & \sqrt{3} \end{pmatrix}\right]$$

Out[ ]= Ld

Es decir, en este ejemplo los vectores son *ld*.

1.4 Vectores en $\mathbb{R}^n$ con VilGebra

En esta sección se comparten algunas herramientas de graficación de vectores en el plano y en el espacio, además de ciertos comandos que permiten resolver problemas de geometría analítica. Si bien es cierto, *Mathematica* facilita funciones de graficación directamente, se han creado ciertas instrucciones en **VilGebra**, con la intención de reducir al máximo el uso de código distractor para el alumno. Como el lector comprobará, con el simple hecho de incluir listas de vectores en los comandos de **VilGebra**, se logra graficar obteniéndose automáticamente los puntos n dimensionales y textos que representan sus correspondientes etiquetas.

1. **GraficaVectores2D**: grafica vectores en el plano, los vectores se añaden como filas de una matriz $n \times 2$. Sintaxis: **GraficaVectores2D[A]**, siendo **A** dicha matriz.

Ejemplo 1.12

Represente en el plano cartesiano los siguientes vectores: $(1,3)$, $\left(-3\sqrt{3},-5\right)$ y $(5,-2)$.

Solución: Utilizando la instrucción **GraficaVectores2D** se resuelve lo solicitado:

```
In[ ]:= GraficaVectores2D[{{1, 3}, {-3√3, -5}, {5, -2}}]
Out[ ]=
```

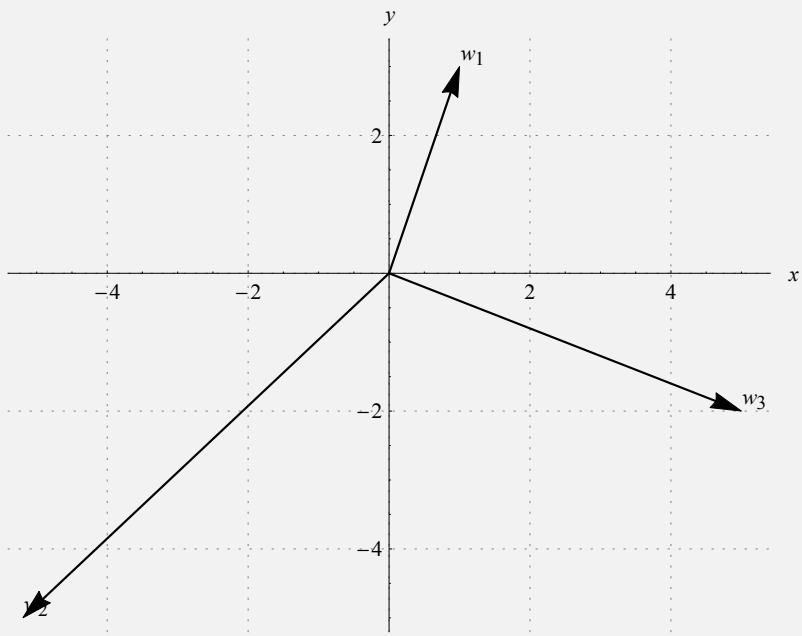

Figura 2: Vectores anclados en el plano

El comando **GraficaVectores2D** auto etiqueta a los vectores ingresados como w_1, w_2, ... dependiendo de la cantidad incluida en la lista que se recibe como parámetro.

2. **GraficaPuntos2D**: grafica puntos en el plano, los vectores se añaden como filas de una matriz $n \times 2$. Sintaxis: **GraficaPuntos2D[A]**, **A** es la matriz contenedora de pares ordenados.

Ejemplo 1.13

Represente en el plano cartesiano los puntos: $(1,3)$, $\left(-3\sqrt{3},-5\right)$ y $(5,-2)$.

Solución: En *Mathematica*:

```
In[ ]:= GraficaPuntos2D[{{1, 3}, {-3√3, -5}, {5, -2}}]
```

```
Out[ ]=
```

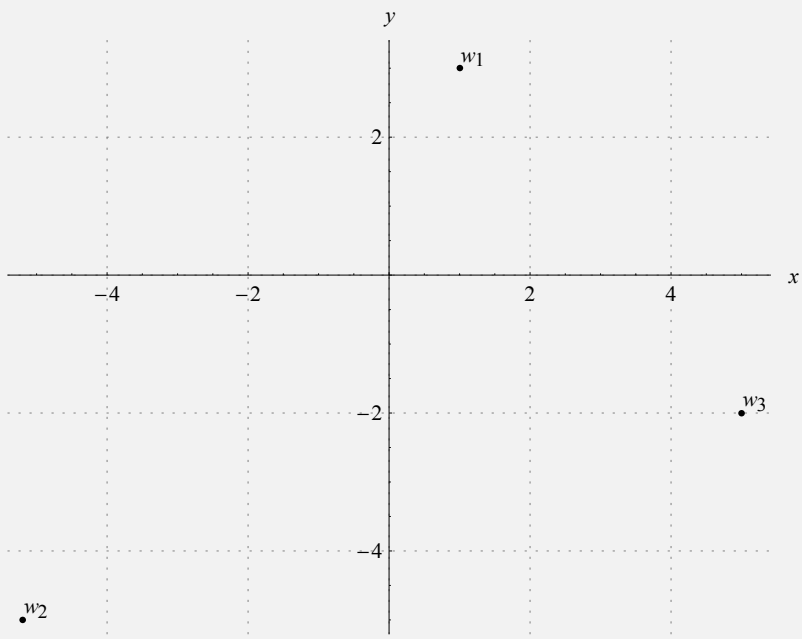

Figura 3: Puntos en el plano

Se diferencia de **GraficaVectores2D** al representar los vectores como puntos del plano.

3. **GraficaVectores3D**: grafica vectores en el espacio tridimensional, los vectores se añaden como filas de una matriz $n \times 3$. Sintaxis: **GraficaVectores3D[A]**, siendo **A** la matriz de vectores en el espacio.

Ejemplo 1.14

Grafique en el espacio los vectores: $(1,2,3)$, $(-2,3,-5)$ y $(5,7,8)$.

Solución: **GraficaVectores3D** produce:

```
In[ ]:= GraficaVectores3D[{{1, 2, 3}, {-2, 3, -5}, {5, 7, 8}}]
```

Out[ ]=

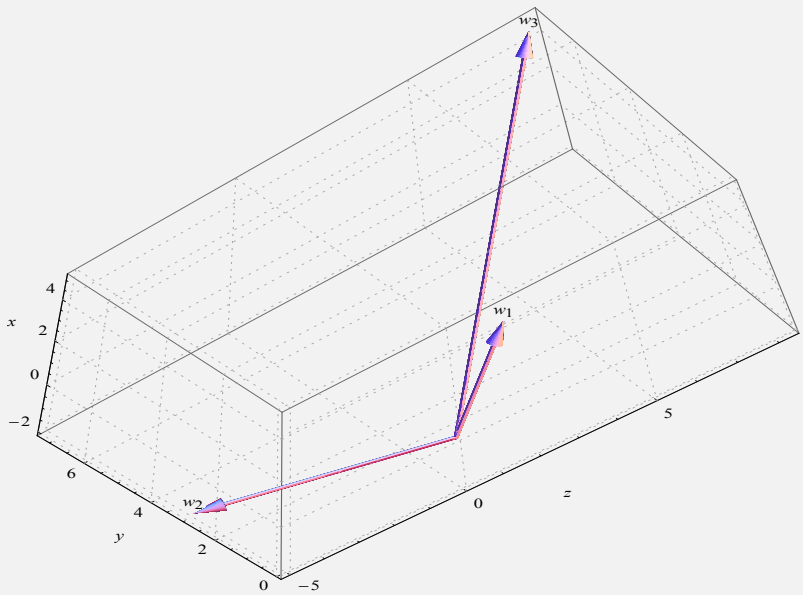

Figura 4: Vectores anclados en el espacio

4. **GraficaPuntos3D**: Grafica puntos en el espacio, los vectores se añaden como filas de una matriz $n \times 3$. Sintaxis: **GraficaPuntos3D[A]**, con **A** la matriz de puntos tridimensionales.

Ejemplo 1.15

Grafique en el espacio los puntos: $(1,2,3)$, $(-2,3,-5)$ y $(5,7,8)$.

Solución: En *Mathematica*:

In[ ]:= GraficaPuntos3D $\left[\begin{pmatrix} 1 & 2 & 3 \\ -2 & 3 & -5 \\ 5 & 7 & 8 \end{pmatrix}\right]$

Out[ ]=

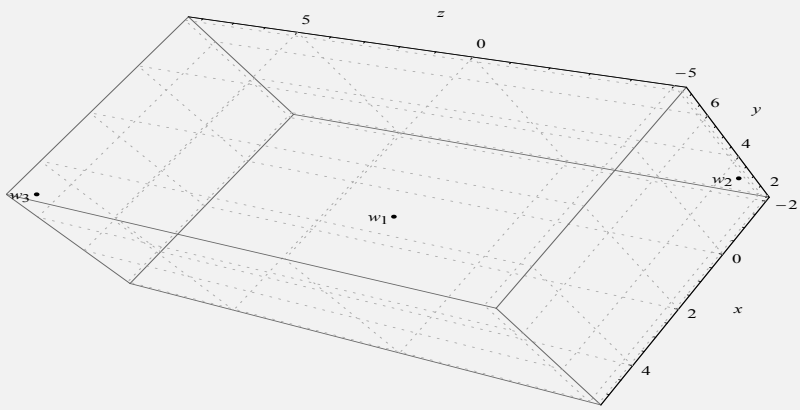

Figura 5: Puntos en el espacio

5. **VectoresParalelosQ**: determina si dos vectores son paralelos retornando True o False según corresponda. Sintaxis: **VectoresParalelosQ[P, Q]**, **P** y **Q** son los vectores que se analizan.

Ejemplo 1.16

Identifique si los siguientes vectores son paralelos: $P = (1,2,3)$ y $Q = (-2,-4,-6)$.

Solución: En el software:

In[ ]:=

$P = \{1,2,3\};$

$Q = \{-2,-4,-6\};$

VectoresParalelosQ$[P,Q]$

Out[ ]= True

El **True** indica que los vectores P y Q sí son paralelos.

Ejercicio

Establezca con ayuda de *Mathematica* si $P \parallel Q$ con: $P = (1,2,3)$ y $Q = (-2,0,-6)$.

6. **ProducMixto**: calcula el producto mixto entre tres vectores con tres componentes. El producto mixto es una combinación entre el producto escalar y el producto cruz en $\mathbb{R}^3$, resolvéndose en primera instancia el producto vectorial. Sintaxis: **ProducMixto[v, w, z]**, lo anterior calcula **v**·**w**×z. Como el lector recordará, por propiedades: **v**·**w**×z=**v**×**w**·z, en cuyo caso el comando **ProducMixto** resuelve cualquiera de estas operaciones.

Ejemplo 1.17

Encuentre $v \times w \cdot z$ donde: $v = (1,-1,2)$, $w = (3,1,-2)$ y $z = (2,2,-1)$.

Solución: En **VilGebra**, la instrucción **ProducMixto** realiza el cálculo:

```
In[ ]:=
 v = {1,-1,2};
 w = {3,1,-2};
 z = {2,2,-1};
 ProducMixto[v,w,z]

Out[ ]= 12
```

7. **AreaParalelogramo2**: calcula el área de un paralelogramo formado por dos vectores. Sintaxis: **AreaParalelogramo2[v, w]**, con **v** y **w** los vectores de $\mathbb{R}^3$ que forman el paralelogramo.

8. **AreaTriangulo2**: calcula el área de un triángulo formado por dos vectores. Sintaxis: **AreaTriangulo2 [v, w]**, donde **v** y **w** son los vectores de $\mathbb{R}^3$ que permiten construir el triángulo.

Ejemplo 1.18

Encuentre el área del paralelogramo y del triángulo formado por: $P = (1,2,3)$ y $Q = (-1,-4,-3)$.

Solución: Como ambas figuras geométricas se generan por **dos** vectores, se usan los comandos **AreaParalelogramo2** y **AreaTriangulo2**, respectivamente:

```
In[ ]:=
 AreaParalelogramo2[{1,2,3},{-1,-4,-3}]
 AreaTriangulo2[{1,2,3},{-1,-4,-3}]

Out[ ]=
 2√10
 √10
```

9. **AreaParalelogramo3**: calcula el área de un paralelogramo formado por tres vectores, uno de ellos funciona como anclado. Sintaxis: **AreaParalelogramo3[v, w, z]**, en esta expresión **v** constituye el vector de anclaje, es decir, la instrucción retorna $\left\| \overrightarrow{\mathbf{vw}} \times \overrightarrow{\mathbf{vz}} \right\|$.

10. **AreaTriangulo3**: calcula el área de un triángulo formado por tres vectores, uno de ellos funciona como anclado. Sintaxis: **AreaTriangulo3[v, w, z]**, en este caso **v** representa el vector de anclaje, al igual que en el comando anterior.

Ejemplo 1.19

Determine el área del paralelogramo y del triángulo construidos a través de: $P=(0,0,0)$, $Q=(1,2,3)$ y $R=(-1,-4,-3)$.

Solución: En este caso se recurre a los comandos **AreaParalelogramo3** y **AreaTriangulo3**, pues son **tres** los vectores dados en el enunciado:

```
In[ ]:=
 AreaParalelogramo3[{0,0,0},{1,2,3},{-1,-4,-3}]
 AreaTriangulo3[{0,0,0},{1,2,3},{-1,-4,-3}]

Out[ ]=
 2√10
 √10
```

11. **VolumenParalelepipedo3**: calcula el volumen de un paralelepípedo formado por tres vectores. Sintaxis: **VolumenParalelepipedo3[v, w, z]**, los vectores **v**, **w** y **z** determinan el paralelepípedo.

12. **VolumenTetraedro3**: calcula el volumen de un tetraedro formado por tres vectores. Sintaxis: **VolumenTetraedro3 [v, w, z]**, **v**, **w** y **z** son puntos tridimensionales que permiten trazar el tetraedro.

Ejemplo 1.20

Halle el volumen del paralelepípedo y del tetraedro construidos por medio de los vectores: $P=(1,-5,2)$, $Q=\left(\sqrt{3},2,3\right)$ y $R=(-1,-4,-3)$.

Solución: En *Mathematica*:

```
In[ ]:=
 VolumenParalelepipedo3[{1,-5,2},{√3,2,3},{-1,-4,-3}]
 VolumenTetraedro3[{1,-5,2},{√3,2,3},{-1,-4,-3}]

Out[ ]=
 -25+23√3
 1/6 (-25+23√3)
```

13. **VolumenParalelepipedo4**: calcula el volumen de un paralelepípedo formado por cuatro vectores. Sintaxis: **VolumenParalelepipedo4[v, w, z, h]**, el vector **v** funciona como el punto de anclaje, de esta manera, la instrucción retorna el resultado de: $\left|\overrightarrow{\mathbf{vw}}\cdot\overrightarrow{\mathbf{vz}}\times\overrightarrow{\mathbf{vh}}\right|$.

14. **VolumenTetrahedro4**: calcula el volumen de un tetraedro formado por cuatro vectores. Sintaxis: **VolumenTetraedro4[v, w, z, h]**, el vector **v** constituye el punto de anclaje para trazar el tetraedro.

Ejemplo 1.21

Encuentre el volumen del paralelepípedo y del tetraedro derivado de los vectores: $P=(0,0,0)$, $Q=(1,-5,2)$, $R=\left(\sqrt{3},2,3\right)$ y $S=(-1,-4,-3)$.

Solución: Al emplear los dos comandos anteriores se obtiene:

```
In[ ]:=
 VolumenParalelepipedo4[{0,0,0},{1,-5,2},{Sqrt[3],2,3},{-1,-4,-3}]
 VolumenTetraedro4[{0,0,0},{1,-5,2},{Sqrt[3],2,3},{-1,-4,-3}]
Out[ ]=
 -25+23 Sqrt[3]
 1/6 (-25+23 Sqrt[3])
```

1.5 Rectas y planos con `VilGebra`

El tema de rectas y planos con frecuencia presenta importantes retos conceptuales al estudiante, principalmente por la dificultad de visualizar conceptos, propiedades y situaciones descritas en los problemas que de manera clásica se desarrollan por parte del docente. Ante este hecho, **`VilGebra`** facilita el uso de ciertas herramientas que con la adecuada mediación pedagógica pueden romper estas brechas cognitivas.

1. **`PerteneceRectaQ`**: determina si un punto Q pertenece a una recta, dado un punto P y su vector director A. Sintaxis: **`PerteneceRectaQ[P, A, Q]`**.

Ejemplo 1.22

Verifique si los vectores $B_1=(0,0,0)$, $B_2=(2,-1,4)$, $B_3=(4,3,-2)$ y $B_4=(2,-9,16)$ pertenecen a la recta de ecuación vectorial: $X=(-3,1,1)+t\,(1,-2,3)$.

Solución: En el software se procede de la siguiente manera:

```
In[ ]:=
P = {-3,1,1};
A = {1,-2,3};
B1 = {0,0,0};
B2 = {2,-1,4};
B3 = {4,3,-2};
B4 = {2,-9,16};
For[i = 1, i ≤ 4, Print[PerteneceRectaQ[P, A, Bi]]; i++]

Out[ ]=
False
False
False
True
```

2. **ColinealesQ**: retorna **True** si tres puntos P, Q y R recibidos como parámetros son colineales y **False** en caso contrario. Sintaxis: **ColinealesQ[P, Q, R]**.

Ejemplo 1.23

Determine si los vectores $P = (-3,1,1)$, $Q = (1,-2,3)$ y R_i son colineales, con: $1 \leq i \leq 2$, $R_1 = (17,-14,11)$ y $R_2 = (2,-9,16)$.

Solución: Al emplear el comando booleano **ColinealesQ**:

```
In[ ]:=
P = {-3,1,1};
Q = {1,-2,3};
R1 = {17,-14,11};
R2 = {2,-9,16};
For[i = 1, i ≤ 2, Print[ColinealesQ[P, Q, Ri]]; i++]

Out[ ]=
True
False
```

3. **EcuacRectaPuntos**: encuentra la ecuación vectorial de una recta dados dos puntos P y Q n-dimensionales, que pertenecen a ella. Sintaxis: **EcuacRectaPuntos[P, Q]**.

Ejemplo 1.24

Halle la ecuación vectorial de la recta que pasa por los puntos: $P=(1,2,3)$ y $Q=(2,4,-4)$.

Solución: Al usar la instrucción EcuacRectaPuntos en *Mathematica*, se obtiene:

```
In[ ]:= EcuacRectaPuntos[{1,2,3},{2,4,-4}]
Out[ ]= {t+1,2(t+1),3-7t}
```

4. **NSecar**: encuentra las coordenadas de los puntos que n-secan a un segmento de recta cuyos extremos son P y Q, en cualquier dimensión. Sintaxis: **NSecar[P, Q, n]**, **n**simboliza el número de partes congruentes en las que se desea n-secar al segmento $\overline{PQ}$.

Ejemplo 1.25

Determine las coordenadas de los puntos que 20-secan al segmento de extremos: $P=(-1,2,5,6,9)$ y $Q=(-7,6,8,9,-9)$.

Solución: Al emplear el comando **NSecar** se encuentran las coordenadas solicitadas:

```
In[ ]:=
 P={-1,2,5,6,9};
 Q={-7,6,8,9,-9};
 NSecar[P,Q,20]
Out[ ]=
```

$\vdots$

$$\mathbf{w}_{14}=\left\{-\frac{26}{5},\frac{24}{5},\frac{71}{10},\frac{81}{10},-\frac{18}{5}\right\}$$

$$\mathbf{w}_{15}=\left\{-\frac{11}{2},5,\frac{29}{4},\frac{33}{4},-\frac{9}{2}\right\}$$

$$\mathbf{w}_{16}=\left\{-\frac{29}{5},\frac{26}{5},\frac{37}{5},\frac{42}{5},-\frac{27}{5}\right\}$$

$$\mathbf{w}_{17}=\left\{-\frac{61}{10},\frac{27}{5},\frac{151}{20},\frac{171}{20},-\frac{63}{10}\right\}$$

$$\mathbf{w}_{18}=\left\{-\frac{32}{5},\frac{28}{5},\frac{77}{10},\frac{87}{10},-\frac{36}{5}\right\}$$

$$\mathbf{w}_{19}=\left\{-\frac{67}{10},\frac{29}{5},\frac{157}{20},\frac{177}{20},-\frac{81}{10}\right\}$$

5. **NSecar2D**: encuentra las coordenadas de los puntos que n-secan a un segmento de recta en el plano con extremos P y Q, y grafica el segmento y los puntos. Sintaxis: **NSecar2D[P, Q, n]**.

Ejemplo 1.26

Determine y grafique los vectores que 20-secan al segmento $\overline{PQ}$ donde: $P=(-1,2)$ y $Q=(-7,6)$.

Solución: En el software:

In[ ]:=

$P=\{-1,2\};$
$Q=\{-7,6\};$
NSecar2D$[P,Q,20]$

Out[ ]=

$$\vdots$$
$$\mathbf{w}_{17}=\left\{-\tfrac{61}{10},\tfrac{27}{5}\right\}$$
$$\mathbf{w}_{18}=\left\{-\tfrac{32}{5},\tfrac{28}{5}\right\}$$
$$\mathbf{w}_{19}=\left\{-\tfrac{67}{10},\tfrac{29}{5}\right\}$$

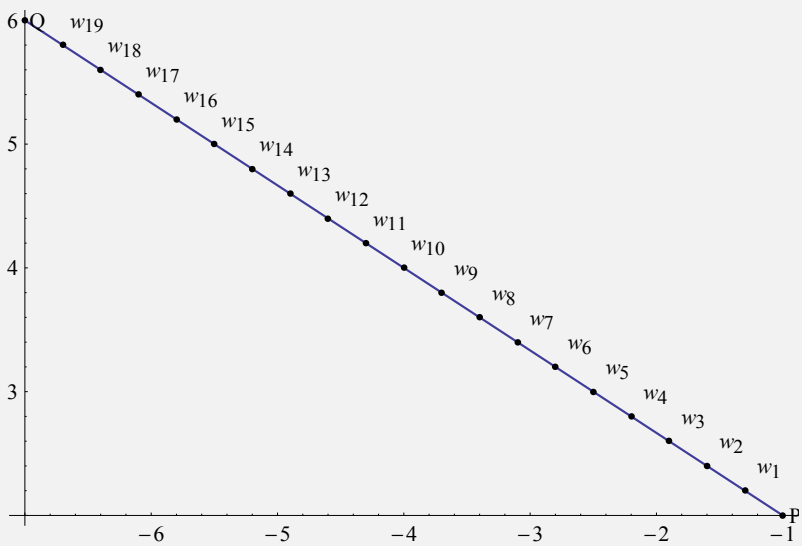

Figura 6: Segmento 20−secado en el plano

6. **NSecar3D**: encuentra las coordenadas de los puntos que n-secan a un segmento de recta en el espacio con extremos P y Q, y grafica el segmento y los puntos. Sintaxis: **NSecar3D[P, Q, n]**.

Ejemplo 1.27

Grafique y determine las coordenadas de los vectores que 20-secan al segmento $\overline{PQ}$ con: $P=(-1,2,1)$ y $Q=(7,6,-11)$.

Solución: **NSecar3D** retorna:

In[ ]:=

$In[]:=$
$P=\{-1,2,1\};$
$Q=\{7,6,-11\};$
NSecar3D$[P,Q,20]$

Out[ ]=

$$\vdots$$
$$\mathbf{w}_{18}=\left\{\frac{31}{5},\frac{28}{5},-\frac{49}{5}\right\}$$
$$\mathbf{w}_{19}=\left\{\frac{33}{5},\frac{29}{5},-\frac{52}{5}\right\}$$

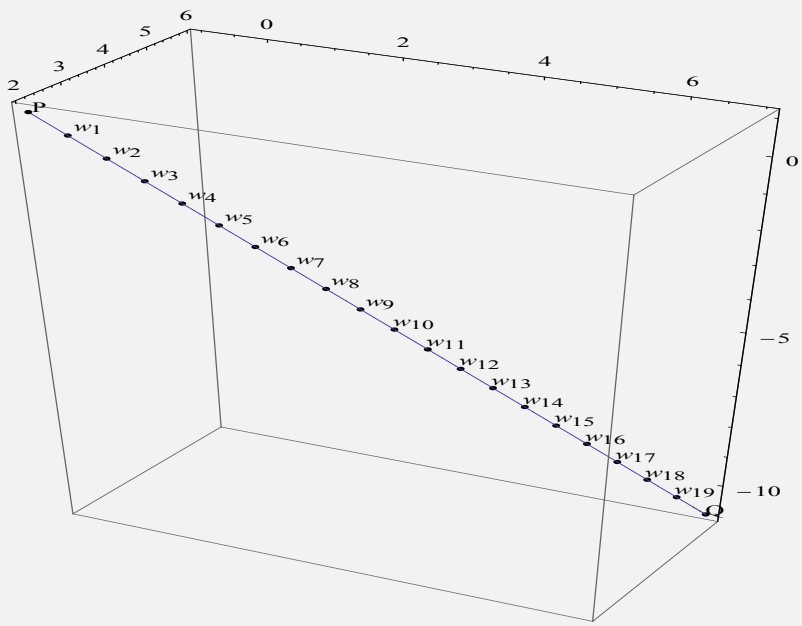

Figura 7: Segmento 20−secado en el espacio

7. **EcuacVecToSime**: encuentra las ecuaciones simétricas o ecuación simétrica de una recta, dada su ecuación vectorial. Sintaxis: **EcuacVecToSime[Ecuaci_List]**, **Ecuaci_List**constituye la ecuación vectorial de la recta.

Ejemplo 1.28

Halle las ecuaciones simétricas de la recta: $X = \left(1+t, 2t-\pi, 3-\frac{7}{3}t\right)$, $t \in \mathbb{R}$.

Solución: En *Mathematica*:

```
In[ ]:= EcuacVecToSime[{t + 1, 2t - π, 3 - 7/3 t}]
```

```
Out[ ]= -1 + x == (Pi + y)/2 == -3/7 (-3 + z)
```

Ejercicio

Dadas las siguientes ecuaciones vectoriales, encuentre las ecuaciones simétricas de cada recta.

(a) $(1+t, 2(1+t), 3-7t)$, $t \in \mathbb{R}$.

(b) $(1, 2(1+t), 3-7t)$, $t \in \mathbb{R}$.

(c) $(1, 2, 3-7t)$, $t \in \mathbb{R}$.

8. **EcuacSimeToVec**: encuentra la ecuación vectorial de una recta dadas sus ecuaciones simétricas o ecuación simétrica. Sintaxis: **EcuacSimeToVec[Ecuaci_List]**, **Ecuaci_List**es un vector que contiene en sus componentes cada una de las expresiones que forman las ecuaciones simétricas de la recta.

Ejemplo 1.29

A través de las ecuaciones simétricas adjuntas, obtenga la ecuación vectorial de la recta correspondiente.

$$\frac{x-1}{\pi} = y - 2 = \frac{z+1}{3}$$

Solución: **EcuacSimeToVec** devuelve en *Mathematica*:

```
In[ ]:= EcuacSimeToVec[{(x - 1)/π, y - 2, (z + 1)/3}]
```

```
Out[ ]= {1 + Pi t, 2 + t, -1 + 3t}
```

Ejercicio

Halle las ecuaciones vectoriales de cada una de las rectas dadas mediante sus ecuaciones simétricas.

(a) $x-1=y-2=\frac{z-7}{8}$

(b) $x-1=y-2,\ z=-7$

(c) $x=6,\ y+1=z-1$

9. **`DistanciaP1Recta`**: determina la distancia de un punto B a una recta dada la recta a través de un punto P y su vector director A. Sintaxis: **`DistanciaP1Recta[B, P, A]`**.

Ejemplo 1.30

Halle la distancia del punto $B=(-3,2,1)$ a la recta $X=P+tA$, $t\in\mathbb{R}$, $P=(2,-1,2)$ y $A=(-1,1,-1)$.

Solución: **`DistanciaP1Recta`** resuelve lo solicitado:

```
In[ ]:=
 In[]:=
 B = {-3,2,1};
 P = {2,-1,2};
 A = {-1,1,-1};
 DistanciaP1Recta[B,P,A]
Out[ ]= 2√2
```

10. **`DistanciaP2Recta`**: determina la distancia de un punto B a una recta dada por medio de dos puntos P y Q. Sintaxis: **`DistanciaP2Recta[B, P, Q]`**.

Ejemplo 1.31

Encuentre la distancia de $B=(-3,2,1)$ a la recta L que pasa por: $P=(2,-1,2)$ y $Q=(1,0,1)$.

Solución: En *Mathematica*:

In[ ]:=

$B = \{-3,2,1\};$
$P = \{2,-1,2\};$
$Q = \{1,0,1\};$
DistanciaP2Recta$[B,P,Q]$

Out[ ]= $2\sqrt{2}$

11. **GraficaRectas2D**: grafica un conjunto de rectas en el plano dadas sus ecuaciones vectoriales en una matriz. Sintaxis: **GraficaRectas2D[Rectas_List, a]**, **Rectas_List**es la matriz de rectas y "**a**" determina el intervalo de graficación **[-a, a]**.

Ejemplo 1.32

Grafique en un mismo sistema de coordenadas las rectas en $\mathbb{R}^2$: $(3+t,1)$, $(1-t,8+2t)$, $\left(\sqrt{3}t,t\right)$, $t \in \mathbb{R}$.

Solución: El comando **GraficaRectas2D** grafica cualquier conjunto de rectas, en este ejemplo:

In[ ]:= **GraficaRectas2D** $\left[\begin{pmatrix} t+3 & 1 \\ 1-t & 2t+8 \\ \sqrt{3}t & t \end{pmatrix}, 10\right]$

Out[ ]=

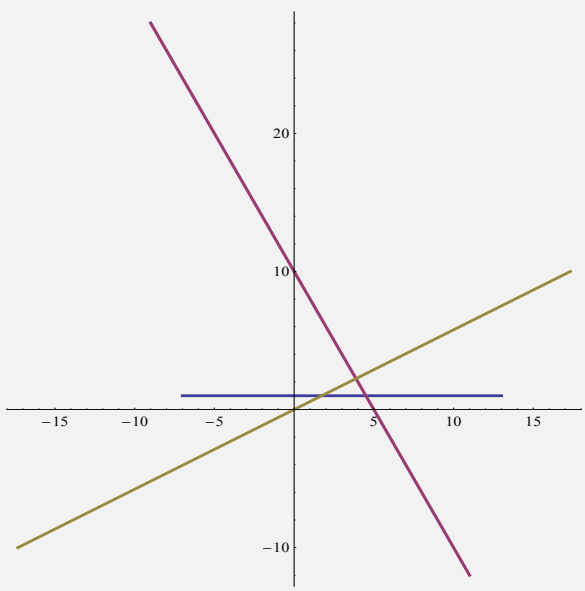

Figura 8: Rectas en el plano

El valor 10 recibido como parámetro del método, define un intervalo de graficación, haciendo variar t de -10 a 10.

12. **`InterRectas2D`**: encuentra el punto de intersección entre dos rectas en el plano cartesiano y grafica el punto y las rectas. Sintaxis: **`InterRectas2D[Rectas_List, s, a]`**, **`Rectas_List`** es una matriz que recibe las ecuaciones vectoriales de cada recta, "**`a`**" determina el intervalo de graficación **`[-a, a]`**, "**`s`**" es un valor bandera, si es igual uno, grafica el punto y las rectas y en cualquier otro caso, no muestra la gráfica, solamente el punto de intersección si es que existe.

Ejemplo 1.33

Determine la intersección entre las rectas de ecuaciones vectoriales: $(1-t, 8+2t)$ y $\left(\sqrt{3}t, t\right)$, $t \in \mathbb{R}$.

Solución: En *Mathematica*:

`In[ ]:= InterRectas2D` $\left[\begin{pmatrix} 1-t & 2t+8 \\ \sqrt{3}t & t \end{pmatrix}, 1, 10\right]$

`Out[ ]=` La intersección de las rectas ocurre en: $\left\{-\frac{10}{11}\left(-6+\sqrt{3}\right), \frac{10}{11}\left(-1+2\sqrt{3}\right)\right\}$

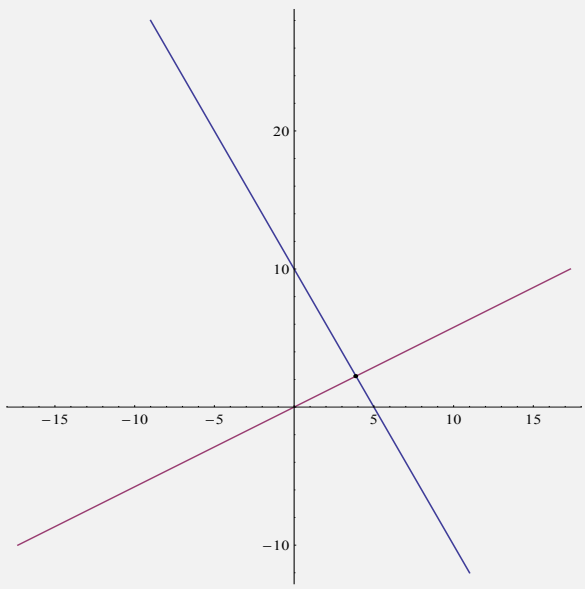

Figura 9: Intersección de dos rectas en el plano

El 1 incluido como parámetro de esta función, le indica a *Mathematica* que grafique las rectas y su punto de intersección. Si el usuario ingresa otro valor distinto de 1, se resuelve el punto de intersección pero no se muestra la gráfica correspondiente.

Ejercicio

Halle la intersección entre las rectas l_1 y l_2.

(a) $\begin{cases} l_1 : (1-t, 8+2t), t \in \mathbb{R} \\ l_2 : (1-2t, 8+4t), t \in \mathbb{R} \end{cases}$

(b) $\begin{cases} l_1 : (1-t, 8+2t), t \in \mathbb{R} \\ l_2 : (4-2t, 6+4t), t \in \mathbb{R} \end{cases}$

13. **`GraficaRectas3D`**: grafica un conjunto de rectas en el espacio dadas sus ecuaciones vectoriales en una matriz **`Rectas_List`**. Sintaxis: **`GraficaRectas3D[Rectas_List, a]`**, "**`a`**" determina el intervalo de graficación **`[-a, a]`**.

Ejemplo 1.34

En $\mathbb{R}^3$ grafique las rectas: $(3+t, 1, -t)$, $(1-t, 8+2t, 1-t)$ y $\left(\sqrt{3}t, t, 0\right)$, $t \in \mathbb{R}$.

Solución: Usando **`GraficaRectas3D`**:

`In[ ]:=` **`GraficaRectas3D`** $\left[\begin{pmatrix} t+3 & 1 & -t \\ 1-t & 2t+8 & 1-t \\ \sqrt{3}t & t & 0 \end{pmatrix}, 10\right]$

`Out[ ]=`

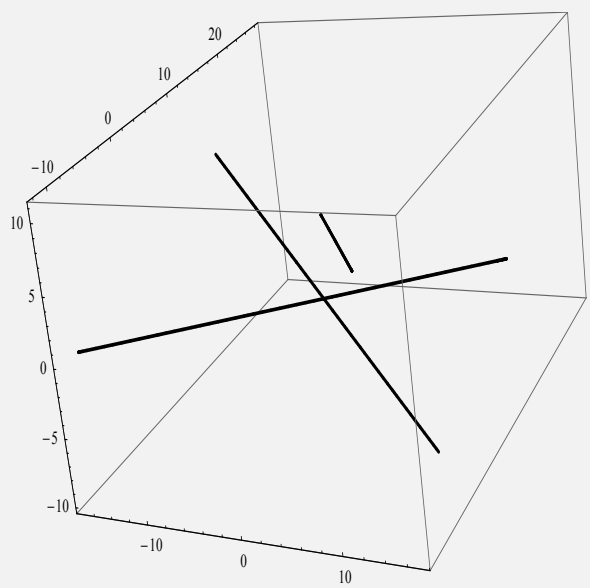

Figura 10: Rectas en el espacio

14. **`InterRectas3D`**: encuentra el punto de intersección entre dos rectas en el espacio y grafica el punto y las rectas. Sintaxis: **`InterRectas3D[Rectas_List, s, a]`**, "**`a`**" determina el intervalo de

graficación **[-a, a]**. Si "**s**" es igual uno grafica el punto y las rectas y en cualquier otro valor, muestra únicamente el punto de intersección.

Ejemplo 1.35

Determine la intersección entre las rectas: $(3+t,5,-t)$ y $(1-t,1-4t,1)$, $t \in \mathbb{R}$.

Solución: En el software se procede así:

In[ ]:= InterRectas3D $\left[\begin{pmatrix} t+3 & 5 & -t \\ 1-t & 1-4t & 1 \end{pmatrix}, 1, 10\right]$

Out[ ]= La intersección de las rectas ocurre en: $\{2,5,1\}$

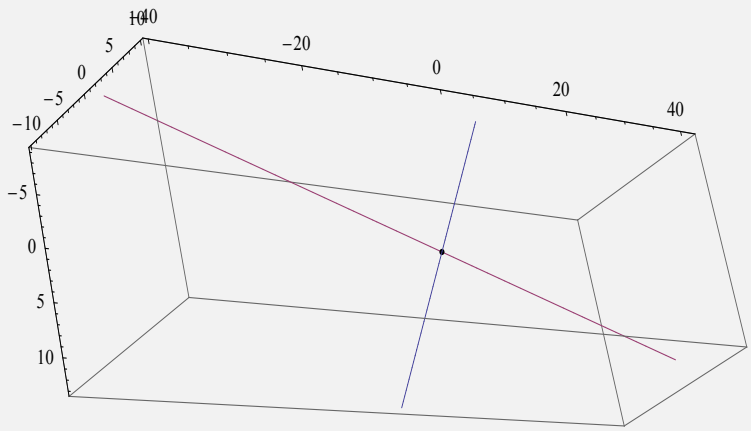

Figura 11: Intersección de dos rectas en el espacio

El lector debe recordar que si se ingresa un valor distinto de 1 como parámetro de la función, ésta encuentra la intersección entre las rectas sin realizar la gráfica.

Ejercicio

Halle la intersección entre las rectas l_1 y l_2.

(a) $\begin{cases} l_1 : (3+t,5,-t), t \in \mathbb{R} \\ l_2 : (4+2t,5,-1-2t), t \in \mathbb{R} \end{cases}$

(b) $\begin{cases} l_1 : (3+t,1,-t), t \in \mathbb{R} \\ l_2 : \left(\sqrt{3}t,t,0\right), t \in \mathbb{R} \end{cases}$

15. **GraficaRectasPuntos2D**: grafica un conjunto de rectas y puntos en el plano cartesiano, dadas sus ecuaciones vectoriales en una matriz **Rectas_List** y la lista de vectores respectivamente. Sintaxis: **GraficaRectasPuntos2D[Rectas_List, Puntos_List, a]**, **Puntos_List** representa la lista de puntos y "**a**" determina el intervalo de graficación **[-a, a]**.

Ejemplo 1.36

Represente en un mismo sistema de coordenadas las rectas: $(3+t,1)$, $(1-t,8+2t)$ y $\left(\sqrt{3}t,t\right)$, $t \in \mathbb{R}$ y los puntos: $(4,1)$, $(18,19)$, $(0,0)$ y $(18,-9)$.

Solución: **GraficaRectasPuntos2D** retorna:

In[ ]:= **GraficaRectasPuntos2D**

$$\left[\begin{pmatrix} t+3 & 1 \\ 1-t & 2t+8 \\ \sqrt{3}t & t \end{pmatrix}, \begin{pmatrix} 4 & 1 \\ 18 & 19 \\ 0 & 0 \\ 18 & -9 \end{pmatrix}, 15\right]$$

Out[ ]=

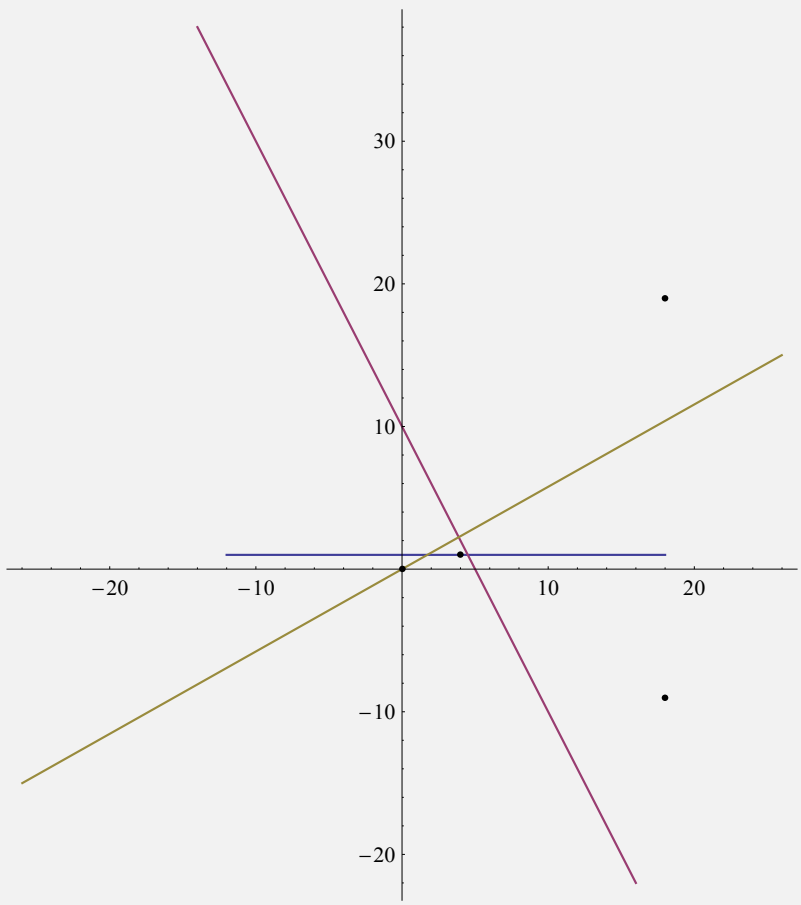

Figura 12: Puntos y rectas en el plano

En el comando el número 15 establece un intervalo de graficación.

16. **GraficaRectasPuntos3D**: grafica un conjunto de rectas y puntos en el espacio, dadas sus ecuaciones vectoriales y la lista de vectores respectivamente. Sintaxis: **GraficaRectasPuntos3D[Rectas_List , Puntos_List, a]**, "**a**" determina el intervalo de graficación **[-a, a]**.

Ejemplo 1.37

Grafique en $\mathbb{R}^3$ las rectas: $(3+t,1,-t)$, $(1-t,8+2t,1-t)$, $\left(\sqrt{3}t,t,0\right)$, $t\in\mathbb{R}$ y los puntos: $(4,1,-1)$, $(18,19,-9)$, $(0,0,0)$ y $(18,-9,3)$.

Solución: Al emplear **GraficaRectasPuntos3D** se obtiene:

In[ ]:= **GraficaRectasPuntos3D**
$$\left[\begin{pmatrix} t+3 & 1 & -t \\ 1-t & 2t+8 & 1-t \\ \sqrt{3}t & t & 0 \end{pmatrix}, \begin{pmatrix} 4 & 1 & -1 \\ 18 & 19 & -9 \\ 0 & 0 & 0 \\ 18 & -9 & 3 \end{pmatrix}, 10\right]$$

Out[ ]=

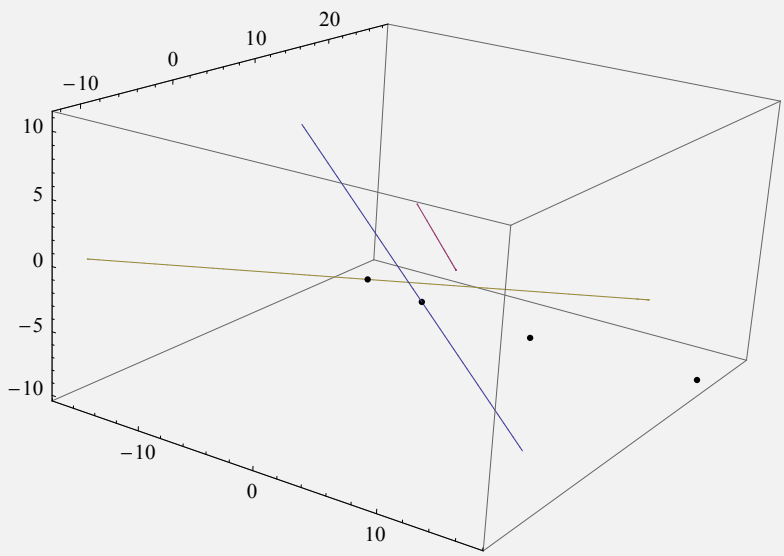

Figura 13: Puntos y rectas en el espacio

17. **PuntoRectaQ**: retorna True si un punto "n" dimensional pertenece a una recta L en dicha dimensión y False en caso contrario. Sintaxis: **PuntoRectaQ[Recta_List, Punto]**, **Recta_List**conforma la ecuación vectorial de la recta y **Punto**el vector que se desea determinar si pertenece o no a la recta dada.

Ejemplo 1.38

Dada la ecuación vectorial $(1+2t,3t,5t,9)$, $t \in \mathbb{R}$, determine si los puntos: $(3,3,5,9)$ y $(3,3,0,9)$ pertenecen a la recta.

Solución: El comando booleano **PuntoRectaQ** soluciona este ejemplo:

```
In[ ]:=
 PuntoRectaQ[{2t + 1,3t,5t,9},{3,3,5,9}]
 PuntoRectaQ[{2t + 1,3t,5t,9},{3,3,0,9}]

Out[ ]=
 True
 False
```

18. **RectaPuntos**: encuentra "n" puntos que pertenecen a una recta dada. Sintaxis: **RectaPuntos[Recta_List, n]**, **Recta_List** es la ecuación vectorial de la recta y **n**la cantidad de puntos que se desean obtener.

Ejemplo 1.39

Determine 10 puntos de la recta con ecuación: $(1+2t,3t,5t,9t,9-t)$, $t \in \mathbb{R}$.

Solución: RectaPuntos en *Mathematica*, puede generar una lista de 10 vectores, todos pertenecientes a la recta:

```
In[ ]:= RectaPuntos[{2t + 1,3t,5t,9t,9 − t},10]

Out[ ]=
 ⋮
 {17,24,40,72,1}
 {19,27,45,81,0}
 {21,30,50,90,−1}
```

19. **GraficaPlanos1**: grafica un conjunto de planos dadas sus ecuaciones cartesianas. Sintaxis: **GraficaPlanos1[Planos_List, a]**, **Planos_List** es una lista de planos con sus ecuaciones cartesianas y "**a**" determina el intervalo de graficación **[-a, a]**.

20. **GraficaPlanos2**: grafica un conjunto de planos dado el sistema de ecuaciones lineales $AX = B$, formado por sus ecuaciones cartesianas. Sintaxis: **GraficaPlanos2[A, B, a]**, "**a**" determina el intervalo de graficación **[-a, a]**.

Ejemplo 1.40

Represente los planos de $\mathbb{R}^3$ cuyas ecuaciones cartesianas son: $x+y=1$, $x+z=2$, $x-y+z=9$ y $x=0$.

Solución: Mediante la instrucción **GraficaPlanos1**:

```
In[ ]:= GraficaPlanos1[{x + y == 1, x + z == 2, x - y + z == 9, x == 0}, 20]

Out[ ]=
```

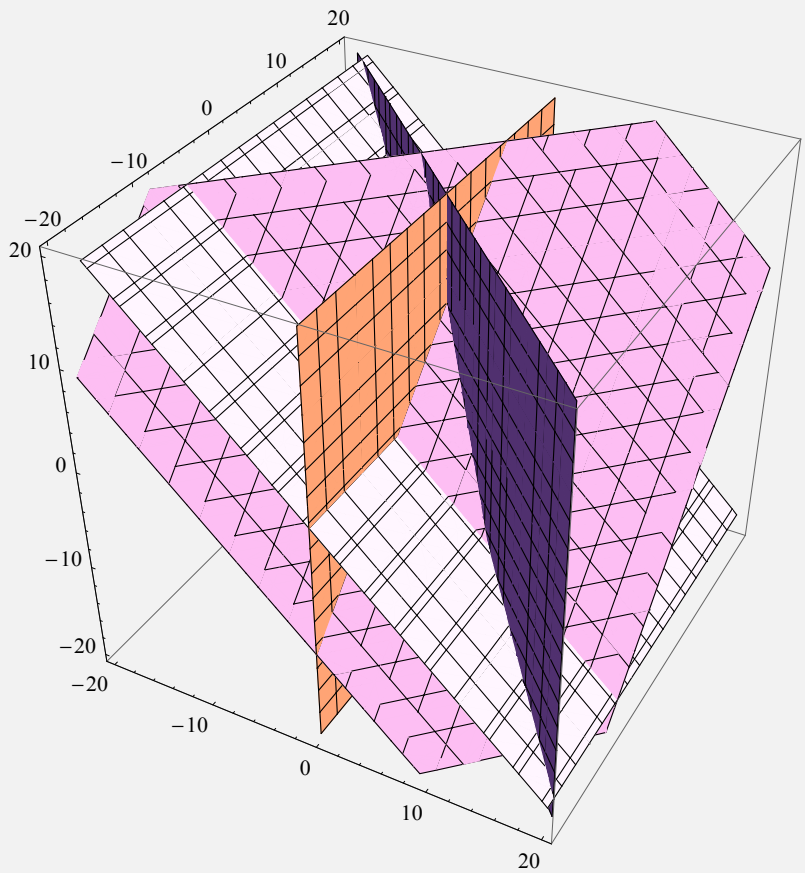

Figura 14: Planos generados por GraficaPlanos1

Utilizando ahora, **GraficaPlanos2**:

In[ ]:=

$$A = \begin{pmatrix} 1 & 1 & 0 \\ 1 & 0 & 1 \\ 1 & -1 & 1 \\ 1 & 0 & 0 \end{pmatrix};$$

$$B = \begin{pmatrix} 1 \\ 2 \\ 9 \\ 0 \end{pmatrix};$$

GraficaPlanos2$[A, B, 20]$

Out[ ]=

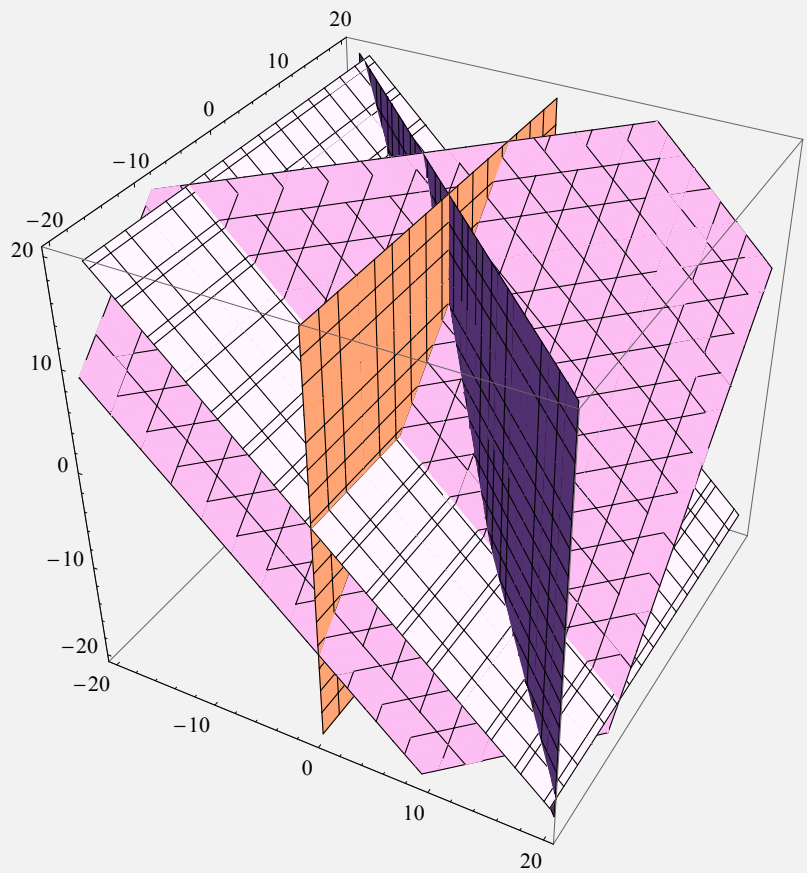

Figura 15: Planos generados por GraficaPlanos2

El valor 20 en ambos comandos, define un intervalo de graficación sobre los ejes coordenados.

21. **PuntoHiperPlanoQ**: retorna **True** si un punto n dimensional **Punto** pertenece a un hiperplano en dicha dimensión y **False** en caso contrario. El lector debe recordar que en $n = 3$ el hiperplano cor-

responde a un plano. La instrucción recibe el hiperplano a través de un vector normal **VNormal** y el término constante **Const** de la ecuación cartesiana. Sintaxis: **PuntoHiperPlanoQ[VNormal, Const, Punto]**.

Ejemplo 1.41

Determine si los vectores: $(0,5,3,2,4)$, $(0,5,3,2,3)$ y $(0,5,3,2)$ pertenecen al hiperplano de ecuación cartesiana: $x+y+z+w+v=14$.

Solución: Recurriendo al uso del comando **PuntoHiperPlanoQ**, se tiene:

```
In[ ]:=
 PuntoHiperPlanoQ[{1,1,1,1,1},14,{0,5,3,2,4}]
 PuntoHiperPlanoQ[{1,1,1,1,1},14,{0,5,3,2,3}]
 PuntoHiperPlanoQ[{1,1,1,1,1},14,{0,5,3,2}]

Out[ ]=
 True
 False
 False
```

22. **HiperPlanoPuntos**: encuentra "n" puntos que pertenecen a un hiperplano dado, el hiperplano se recibe a través de su ecuación cartesiana contenida en el vector **Plano_List**. Sintaxis: **HiperPlanoPuntos [Plano_List, n]**.

Ejemplo 1.42

Determine 10 puntos del hiperplano $5x+2y-z-w+6v=14$.

Solución: En *Mathematica*:

```
In[ ]:= HiperPlanoPuntos[{6v − w + 5x + 2y − z == 14},10]

Out[ ]=
 {v,w,x,y,z} =
 {3/2,1,1,1,1}
 ⋮
 {−31/6,9,9,9,9}
 {−6,10,10,10,10}
```

Ejercicio

Halle 10 puntos del hiperplano con ecuación cartesiana: $5x+2y-z-w+6v-p-9a=20$.

23. **GraficaPlanosPuntos**: grafica un conjunto planos y puntos, dadas sus ecuaciones cartesianas y la lista de vectores respectivamente. Sintaxis: **GraficaPlanosPuntos[Planos_List, Puntos_List, a]** , "**a**" determina el intervalo de graficación **[-a, a]**.

Ejemplo 1.43

Represente en $\mathbb{R}^3$ y en un mismo sistema de coordenadas, los planos: $x+y=1$, $x=0$ y los vectores: $(4,1,-1)$, $(18,19,-9)$, $(0,0,0)$, $(18,-9,3)$.

Solución: La instrucción de **VilGebra**, **GraficaPlanosPuntos** resuelve lo solicitado:

In[ ]:= **GraficaPlanosPuntos** $\left[\{x+y==1, x==0\}, \begin{pmatrix} 4 & 1 & -1 \\ 18 & 19 & -9 \\ 0 & 0 & 0 \\ 18 & -9 & 3 \end{pmatrix}, 30\right]$

Out[ ]=

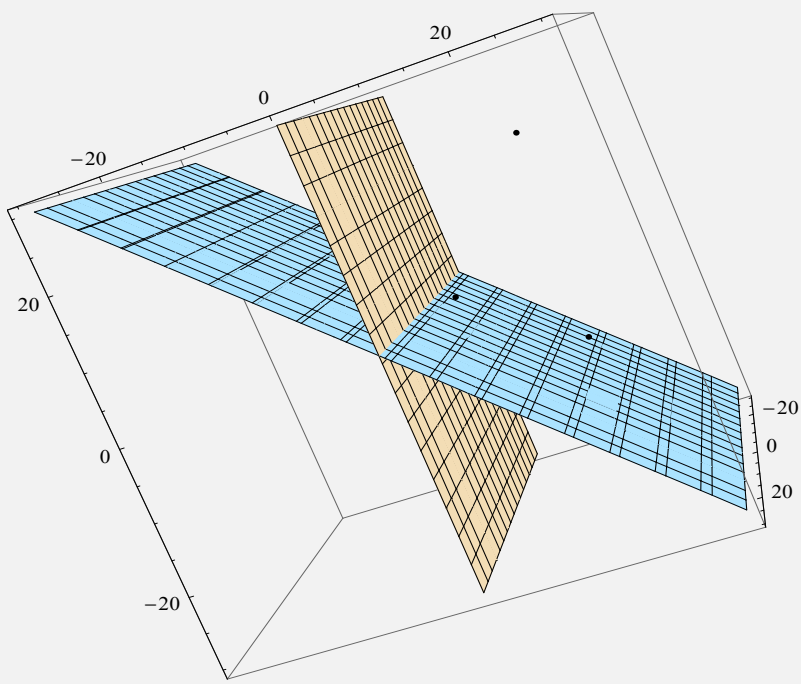

Figura 16: Puntos y planos en el espacio

El valor 30 ingresado como último parámetro del método, define un intervalo de graficación sobre los tres ejes "x", "y" y "z".

24. **`GraficaRectasPlanos`**: grafica un conjunto de rectas y planos dadas sus ecuaciones vectoriales y cartesianas, respectivamente. Sintaxis: **`GraficaRectasPlanos[Rectas_List, Planos_List, a]`**, "**`a`**" determina el intervalo de graficación **`[-a, a]`**.

Ejemplo 1.44

Grafique en un mismo sistema de coordenadas, las rectas de ecuaciones: $(3+t,1,-t)$, $(1-t,8+2t,1-t)$, $\left(\sqrt{3}t,t,0\right)$, $t\in\mathbb{R}$ y los planos: $x+y=1$, $x+z=2$, $x-y+z=9$, $x=0$.

Solución: En el software:

```
In[ ]:=
```

$$\texttt{GraficaRectasPlanos}\left[\begin{pmatrix} t+3 & 1 & -t \\ 1-t & 2t+8 & 1-t \\ \sqrt{3}t & t & 0 \end{pmatrix},\right.$$
$$\left.\{x+y==1, x+z==2, x-y+z==9, x==0\}, 30\right]$$

Out[ ]=

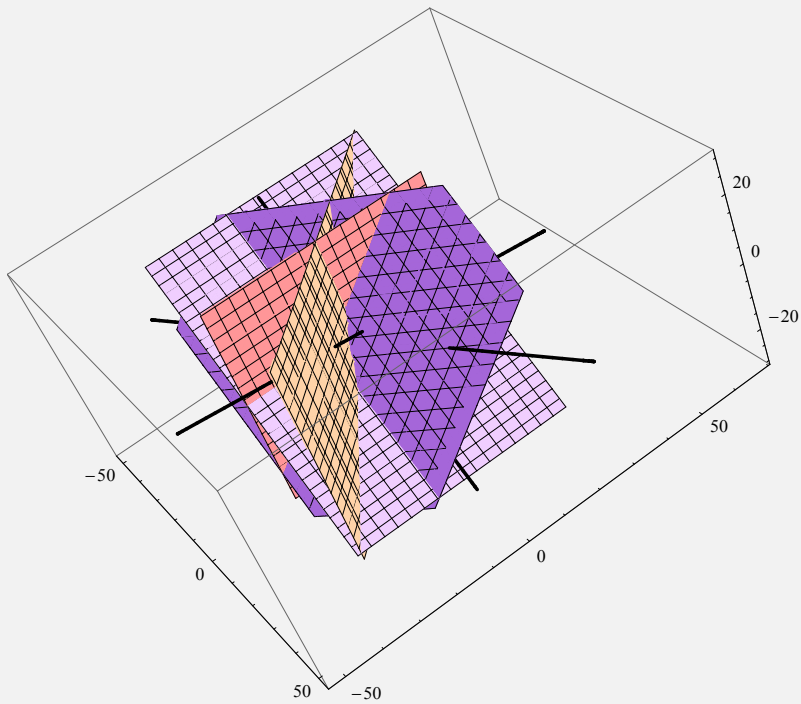

Figura 17: Rectas y planos en el espacio

Al igual que en la instrucción anterior, el valor 30 al final del comando, define un intervalo de graficación.

25. **GraficaRectasPlanosPuntos**: grafica un conjunto de rectas, planos y puntos, dadas sus ecuaciones vectoriales, cartesianas y la lista de vectores respectivamente. Sintaxis: **GraficaRectasPlanosPuntos** [
Rectas_List, **Planos_List**, **Puntos_List**, **a**], "**a**" determina el intervalo de graficación **[-a, a]**.

Ejemplo 1.45

Represente simultáneamente en $\mathbb{R}^3$ las rectas: $(3+t,1,-t)$, $(1-t,8+2t,1-t)$, $\left(\sqrt{3}t,t,0\right)$, $t \in \mathbb{R}$, los planos: $x+y=1$, $x+z=2$, $x-y+z=9$, $x=0$ y los vectores: $(4,1,-1)$, $(18,19,-9)$, $(0,0,0)$, $(18,-9,3)$.

Solución: En *Mathematica*:

In[ ]:=

GraficaRectasPlanosPuntos $\left[\begin{pmatrix} t+3 & 1 & -t \\ 1-t & 2t+8 & 1-t \\ \sqrt{3}t & t & 0 \end{pmatrix},\right.$
$\{x+y==1, x+z==2, x-y+z==9, x==0\},$
$\left.\begin{pmatrix} 4 & 1 & -1 \\ 18 & 19 & -9 \\ 0 & 0 & 0 \\ 18 & -9 & 3 \end{pmatrix}, 10\right]$

Out[ ]=

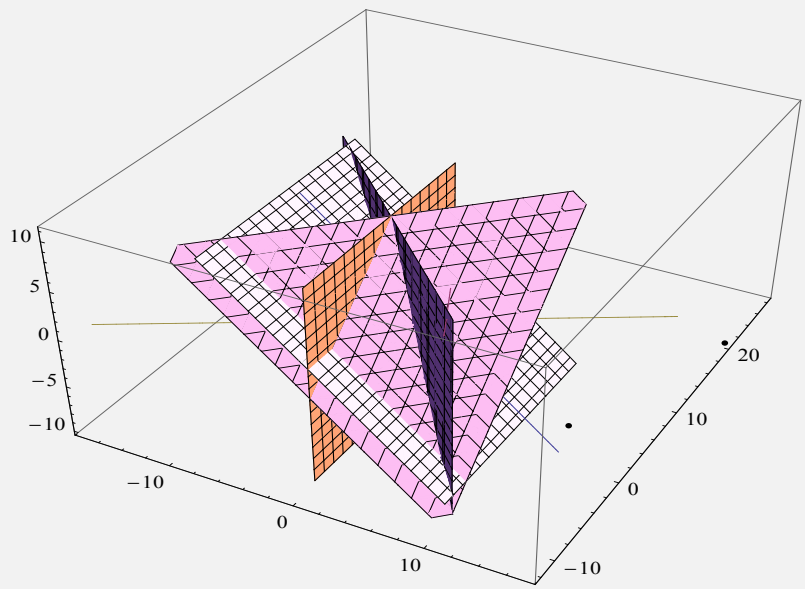

Figura 18: Puntos, rectas y planos en el espacio

26. **EcuacCartPuntoGene**: retorna la ecuación cartesiana de un plano en el espacio, conociendo un punto P y sus vectores generadores A y B. Sintaxis: **EcuacCartPuntoGene[P, A, B]**.

Ejemplo 1.46

Determine la ecuación cartesiana de un plano que pasa por $P=(1,1,-1)$ y cuyos vectores generadores son $A=(-3,3,3)$ y $B=(0,-2,3)$.

Solución: El comando **EcuacCartPuntoGene** retorna la ecuación buscada:

```
In[ ]:=
 P = {1,1,-1};
 A = {-3,3,3};
 B = {0,-2,3};
 EcuacCartPuntoGene[P, A, B]
Out[ ]= 5x + 3y + 2z == 6
```

27. **EcuacCartPuntos**: retorna la ecuación cartesiana de un plano en el espacio, conociendo tres puntos del plano P, Q y R no colineales. Sintaxis: **EcuacCartPuntos[P, Q, R]**.

Ejemplo 1.47

Se sabe que un plano en $\mathbb{R}^3$ contiene a los puntos $P = (2,1,3)$, $Q = (-1,-2,4)$ y $R = (4,2,1)$. Encuentre su ecuación cartesiana.

Solución: En *Mathematica* se procede así:

```
In[ ]:=
 P = {2,1,3};
 Q = {-1,-2,4};
 R = {4,2,1};
 EcuacCartPuntos[P, Q, R]
Out[ ]= -15 + 5x + 3z == 4y
```

EcuacCartPuntos verifica que los vectores ingresados no sean colineales. Si lo fueran, imprime un mensaje a este respecto.

Ejercicio

Un plano contiene a los vectores de $\mathbb{R}^3$: $P = (2,1,3)$, $Q = (-1,-2,4)$ y $R = (-1,-2,4)$. Halle si es posible su ecuación cartesiana.

28. **RectaPlanos**: determina la ecuación vectorial de una recta de intersección entre dos planos en la dimensión 3 y dibuja la recta y los planos. Sintaxis: **RectaPlanos[Planos_List, s, a]**, si no se desea visualizar la gráfica se le da a "**s**" un valor distinto de 1. Además, "**a**" establece el intervalo de graficación **[-a, a]**.

Ejemplo 1.48

Encuentre la ecuación de la recta de intersección entre los planos $2x+3y+z=5$ y $x-3y-2z=1$. Grafique los planos y la recta resultante.

Solución: **`RectaPlanos`** retorna:

```
In[ ]:= RectaPlanos[{2x + 3y + z == 5, x - 3y - 2z == 1}, 1, 20]

Out[ ]=
```

La ecuación vectorial de la recta de intersección de los planos es:$\{2-3t, 1/3+5t, -9t\}$

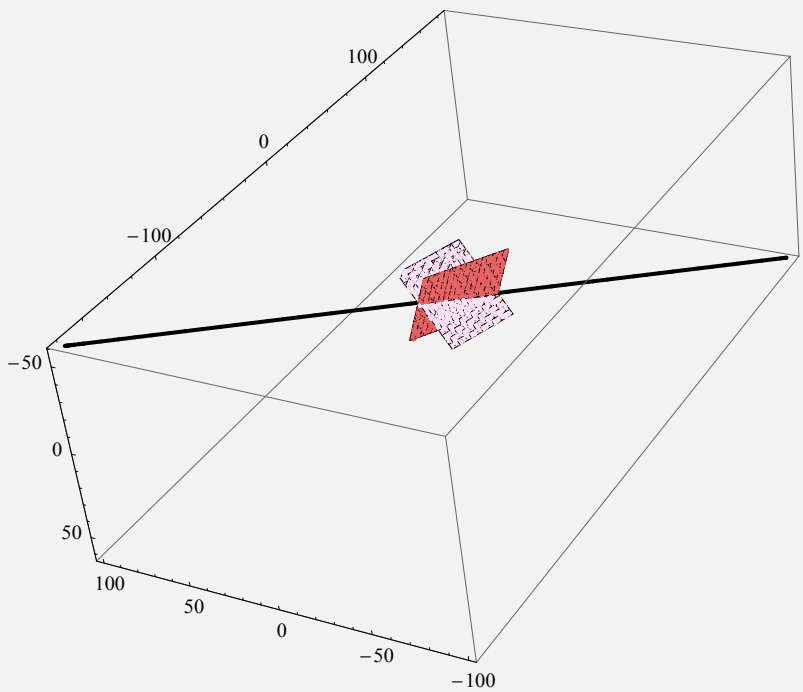

Figura 19: Recta generada por dos planos

El 1 añadido como parámetro del método, indica a *Mathematica* realizar la gráfica correspondiente, tomando como base el número 20 que se especificó como último dato. Si el usuario no desea graficar, basta con tomar un valor real distinto de 1. También es importante aclarar al lector, que RectaPlanos devuelve un mensaje sin realizar ningún cálculo, en caso de tener dos planos paralelos.

Ejercicio

Encuentre la intersección entre los planos: $2x+3y+z=5$ y $2x+3y+z=1$.

29. **DistanciaPuntoPlano**: determina la distancia de un punto P a un plano en la tercera dimensión. La ecuación debe ser igualada a un término constante y expresada en términos de "x", "y" y "z". Sintaxis: **DistanciaPuntoPlano[Planos_List, P]**, **Planos_List** es un vector que contiene la ecuación cartesiana del plano.

Ejemplo 1.49

Halle la distancia del punto $(2,5,-1)$ al plano de ecuación cartesiana $2x+6y-z=9$.

Solución: **DistanciaPuntoPlano** calcula la distancia pedida:

In[ ]:= DistanciaPuntoPlano$[\{2x+6y-z==9\},\{2,5,-1\}]$

Out[ ]= $\frac{26}{\sqrt{41}}$

30. **DistanciaPlanos**: encuentra la distancia entre dos planos paralelos. Las ecuaciones deben ser igualadas a un término constante y expresadas en términos de "x", "y" y "z" Sintaxis: **DistanciaPlanos[Planos_List]**, **Planos_List**es una lista cuyas componentes son las ecuaciones cartesianas de los planos paralelos.

Ejemplo 1.50

Determine la distancia entre los planos: $2x+3y-z=1$ y $4x+6y-2z=3$.

Solución: En *Mathematica*:

In[ ]:= DistanciaPlanos$[\{2x+3y-z==1,4x+6y-2z==3\}]$

Out[ ]= $\frac{1}{2\sqrt{14}}$

Si los planos no son paralelos, o bien, los parámetros de la instrucción corresponden a hiperplanos en la dimensión mayor a tres, **DistanciaPlanos** lo indica al usuario.

Ejercicio

Halle si es posible, la distancia entre los planos con ecuaciones cartesianas:

(a) $2x+3y-z=1$ y $x+6y-2z=3$

(b) $2x+3y-z-v=1$ y $x+6y-2z=3$

1.6 Espacios vectoriales con `VilGebra`

El tema de espacios vectoriales es por naturaleza teórico-abstracto, esto lo hace parecer imposible de automatizar en algunas de sus áreas de conocimiento, mediante rutinas de programación. Pese a ello, se comparten dos comandos integrados en **`VilGebra`**: el primero facilita la búsqueda seudoaleatoria de una base en un espacio vectorial de dimensión finita, partiendo de una familia libre y el segundo determina las coordenadas de un vector, dada una base del espacio vectorial.

1. **`CompletandoBase`**: completa una familia libre para formar de manera seudoaleatorio una base de un espacio vectorial de dimensión finita. Sintaxis: **`CompletandoBase[A, n]`**, **`A`** es una matriz cuyas filas contienen cada uno de los elementos de la familia libre y **`n`** representa la dimensión del espacio.

Ejemplo 1.51

En el espacio vectorial de polinomios de grado menor o igual a seis, denotado $\mathbb{R}_6[x]$, halle una base B que contenga a la familia libre $X = \{x + x^3, x^2 + x^6, 1 + x - x^3\}$.

Solución: Los elementos de X se pueden almacenar en una matriz, siendo así, en *Mathematica* y mediante el uso del comando **`CompletandoBase`**, se obtiene:

```
In[ ]:=
```

$$A = \begin{pmatrix} 0 & 0 & 0 & 1 & 0 & 1 & 0 \\ 1 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & -1 & 0 & 1 & 1 \end{pmatrix};$$

```
CompletandoBase[A,7]
```

`Out[ ]=` La base buscada se encuentra contenida en la matriz:

$$\begin{pmatrix} 0 & 0 & 0 & 1 & 0 & 1 & 0 \\ 1 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & -1 & 0 & 1 & 1 \\ 0 & 1 & 0 & 0 & 1 & 1 & 1 \\ 0 & 1 & 1 & 1 & 0 & 1 & 1 \\ 1 & 1 & 1 & 1 & 0 & 0 & 1 \\ 0 & 1 & 1 & 1 & 0 & 1 & 0 \end{pmatrix}$$

Naturalmente el lector analizará que el segundo parámetro de CompletandoBase (en este ejemplo, el número siete), corresponde a la dimensión del espacio vectorial. La matriz devuelta permite concluir que:

$$B = \{x + x^3, x^2 + x^6, 1 + x - x^3, x^5 + x^2 + x + 1, \\ x^5 + x^4 + x^3 + x + 1, x^6 + x^5 + x^4 + x^3 + 1, x^5 + x^4 + x^3 + x\}$$

Paquete ***`VilGebra`*** . Enrique Vílchez Q.

Lo interesante del comando **CompletandoBase**, radica en su capacidad de generar cada vez una base distinta que contiene la familia libre inicial, es decir, si el usuario ejecuta la instrucción varias veces sin cambiar sus parámetros, observará bases distintas del espacio vectorial.

2. **VectorCoordenadas**: determina el vector de coordenadas de **x**, recibiendo la base **B** correspondiente y la dimensión del espacio vectorial. Sintaxis: **VectorCoordenadas[x, B, Dimension]**.

Ejemplo 1.52

Halle el vector de coordenadas de (x,y,z) con respecto a $B=\{(5,2,-1),(0,-1,2),(3,0,5)\}$ en $\mathbb{R}^3$.

Solución: En el software:

```
In[ ]:= VectorCoordenadas[{x,y,z}, {{5, 2, -1}, {0, -1, 2}, {3, 0, 5}}, 3]
```

$$\text{Out[ ]:=}\begin{pmatrix} \frac{1}{16}(5x-6y-3z) \\ \frac{5x}{8}-\frac{7y}{4}-\frac{3z}{8} \\ -\frac{3x}{16}+\frac{5y}{8}+\frac{5z}{16} \end{pmatrix}$$

El 3 ingresado como último parámetro del comando **VectorCoordenadas**, representa la dimensión del espacio vectorial $\mathbb{R}^3$. **VectorCoordenadas** verifica que el conjunto B sea una base del espacio, si no es así, lo indica al usuario.

Ejercicio

Encuentre el vector de coordenadas en cada caso, de acuerdo con la base B dada.

(a) $v=(3,4)$, $B=\{(-1,1),(0,2)\}$ con respecto a $\mathbb{R}^2$.

(b) $v=(0,8,32)$, $B=\{(5,2,-1),(0,-1,2),(3,0,5)\}$ con respecto a $\mathbb{R}^3$.

1.7 Proyecciones ortogonales con VilGebra

Los cálculos implicados en una proyección ortogonal muchas veces involucran operaciones exhaustivas en las cuales con frecuencia los alumnos suelen cometer errores de carácter aritmético, por esta razón,

en **VilGebra** se han incluido tres comandos que permiten revisar paso a paso este tipo de procedimientos, entre ellos: el método de *Gram Schmidt*, el cálculo de una proyección ortogonal y el cálculo del complemento ortogonal a un subespacio de $\mathbb{R}^n$.

1. **MGS**: aplica paso a paso el método de *Gram Schmidt*. Sintaxis: **MGS[B]**, **B** es una matriz cuyas filas forman una base del subespacio de $\mathbb{R}^n$ correspondiente.

Ejemplo 1.53

Halle una base ortonormal para el subespacio vectorial de $\mathbb{R}^4$:

$$W = Gen\{(1,1,0,0),(0,2,2,0),(0,0,3,3)\}$$

Solución: **MGS** retorna:

In[ ]:=

$$\text{MGS}\left[\begin{pmatrix} 1 & 1 & 0 & 0 \\ 0 & 2 & 2 & 0 \\ 0 & 0 & 3 & 3 \end{pmatrix}\right]$$

Out[ ]=

Vector 1 de la base ortonormal: $\left\{\frac{1}{\sqrt{2}},\frac{1}{\sqrt{2}},0,0\right\}$

Vector 2 de la base ortonormal: $\left\{-\frac{1}{\sqrt{6}},\frac{1}{\sqrt{6}},\sqrt{\frac{2}{3}},0\right\}$

Con i=2, $\text{Proy}_{S_{i-1}}U_i = \{1,1,0,0\}$ y $U_i - \text{Proy}_{S_{i-1}}U_i = \{-1,1,2,0\}$

Vector 3 de la base ortonormal: $\left\{\frac{1}{2\sqrt{3}},-\frac{1}{2\sqrt{3}},\frac{1}{2\sqrt{3}},\frac{\sqrt{3}}{2}\right\}$

Con i=3, $\text{Proy}_{S_{i-1}}U_i = \{-1,1,2,0\}$ y $U_i - \text{Proy}_{S_{i-1}}U_i = \{1,-1,1,3\}$

$$\begin{pmatrix} \frac{1}{\sqrt{2}} & \frac{1}{\sqrt{2}} & 0 & 0 \\ -\frac{1}{\sqrt{6}} & \frac{1}{\sqrt{6}} & \sqrt{\frac{2}{3}} & 0 \\ \frac{1}{2\sqrt{3}} & -\frac{1}{2\sqrt{3}} & \frac{1}{2\sqrt{3}} & \frac{\sqrt{3}}{2} \end{pmatrix}$$

La última matriz contiene en sus filas la base ortonormal buscada. **MGS** verifica que el parámetro ingresado corresponda a un conjunto de vectores linealmente independientes, de lo contrario, no realiza ningún cálculo, indicando al usuario el error en la base incluida. El lector debe saber que *Mathematica* por defecto posee el comando **Orthogonalize**, el cuál permite también, hallar bases ortonormales, sin mostrar el proceso paso a paso. El acrónimo **MGS** significa "Método de *Gram Schmidt*".

2. **PrOt**: calcula la proyección ortogonal de un vector **x**sobre un subespacio de $\mathbb{R}^n$. Sintaxis: **PrOt[x , B]**, **B** es una base cualquiera del subespacio (en formato de matriz). **B**no necesariamente debe ser ortonormal, pues la rutina internamente ortonormaliza.

Ejemplo 1.54

Calcule la proyección ortogonal de $v = (0,1,2,3)$ con respecto a:

$$B = \{(1,1,0,0),(0,2,2,0),(0,0,3,3)\}$$

Solución: En *Mathematica*:

```
In[ ]:= PrOt[{0,1,2,3},{{1,1,0,0},{0,2,2,0},{0,0,3,3}}]
```

```
Out[ ]= {1/2, 1/2, 5/2, 5/2}
```

PrOt al igual que **MGS**, corrobora que B sea un conjunto de vectores li. Como se observa en este ejemplo, no es necesario que B sea una base ortonormal, pues la instrucción ortonormaliza internamente.

3. **ComplementOt**: calcula el complemento de un vector **x** ortogonal a un subespacio de $\mathbb{R}^n$. Sintaxis: **ComplementOt[x, B]**, **B**es una base cualquiera del subespacio.

Ejemplo 1.55

Determine el complemento de $v = (0,1,2,3)$ ortogonal al subespacio de $\mathbb{R}^4$:

$$W = Gen\{(1,1,0,0),(0,2,2,0),(0,0,3,3)\}$$

Solución: **ComplementOt** resuelve lo solicitado en este ejemplo:

```
In[ ]:= ComplementOt[{0,1,2,3}, {{1,1,0,0},{0,2,2,0},{0,0,3,3}}]
```

```
Out[ ]= {-1/2, 1/2, -1/2, 1/2}
```

El método también verifica si la base ingresada, constituye una familia libre de vectores.

1.8 Aplicaciones lineales con VilGebra

Las transformaciones lineales es un tema que no se aborda directamente en el software *Mathematica*, de allí la necesidad de programar distintas funciones para resolver procedimientos típicos, tales

como: determinar si una aplicación es lineal, hallar el núcleo, hallar la imagen, encontrar la matriz que representa la transformación, entre otros. Este es el objetivo de la presente sección, compartir algunas herramientas de cálculo que automatizan ciertos algoritmos usualmente resueltos de manera tradicional, sin apoyo de software.

1. **TLQ**: devuelve **True** si dada una función ella constituye una transformación lineal y **False** en caso contrario, si la función no es aplicación lineal brinda un contraejemplo. Sintaxis: **TLQ[T, n, cero]**, **T** es el criterio de asociación de la aplicación donde la preimagen es un vector de $\mathbb{R}^n$, "n" es la dimensión del espacio vectorial dominio y **cero** es el cero del espacio vectorial codominio.

Ejemplo 1.56

Utilizando el comando **TLQ** demuestre si T es una aplicación lineal, con $T:\mathbb{R}_1[x]\to\mathbb{R}_2[x]$, tal que: $T(at+b)=t(at+b)$.

Solución: En el software:

```
In[ ]:=
 T[{a_,b_}]:=t(at+b)
 TLQ[T,2,0]

Out[ ]= True
```

Se concluye que T sí es una transformación lineal. El lector debe comprender que el comando **TLQ** recibe como parámetros la transformación lineal T (que idealmente debe ser escrita como una función de $\mathbb{R}^n\to\mathbb{R}^m$), la dimensión del espacio vectorial dominio y el cero del espacio vectorial codominio. Si la preimagen de T no se expresa como un vector de $\mathbb{R}^n$ la instrucción **TLQ** no ejecuta correctamente.

Ejemplo 1.57

Sea $T:M_2(\mathbb{R})\to M_2(\mathbb{R})$, tal que: $T(A)=A^t$. Determine si T es una transformación lineal.

Solución: En *Mathematica*:

```
In[ ]:=
 T[{x_,y_,z_,w_}]:={x,z,y,w}
 TLQ[T,4,{0,0,0,0}]

Out[ ]= True
```

El **True** indica que T sí es aplicación lineal.

Ejemplo 1.58

Si $T:\mathbb{R}^2 \to \mathbb{R}$ con $T(x,y) = |x-y|$, ¿T es una aplicación lineal?

Solución: Al emplear **TLQ**:

```
In[ ]:=
T[{x_,y_}]:=Abs[x - y]
TLQ[T,2,0]

Out[ ]=
False
Un contraejemplo ocurre en el vectorv = {0,1}, el vectorw = {13,11},
el escalar α = 16 y el escalar β = 12
```

En este ejemplo T no es una transformación lineal. Es interesante observar cómo si **TLQ** nos retorna **False**, al mismo tiempo genera de forma seudoaleatoria un contraejemplo. Lo anterior significa, que al correr **TLQ** sin cambiar ningún parámetro del método, generará cada vez un ejemplo distinto de no linealidad.

Ejercicio

Establezca si las funciones dadas son transformaciones lineales.

(a) $T:\mathbb{R}^2 \to \mathbb{R}$ con $T(x,y) = \text{Norm}[(x,y)]$.

(b) $T:\mathbb{R}^3 \to \mathbb{R}^3$ con $T(x,y,z) = (x,y,2-x-y)$.

2. **Nucleo**: determina como un conjunto generado el núcleo de una transformación lineal de $\mathbb{R}^n$ a $\mathbb{R}^m$. Además, indica si la aplicación es inyectiva. Sintaxis: **Nucleo[T, n, cero]**, los argumentos que recibe son iguales a los del comando **TLQ**.

Ejemplo 1.59

Sea la transformación lineal $T:M_2(\mathbb{R}) \to \mathbb{R}^2$, tal que:

$$T\begin{pmatrix} x & y \\ z & w \end{pmatrix} = \left(\frac{-13x+8y-9z+w}{3}, -3x+2y-2z+w\right)$$

Encuentre el núcleo, ¿es T inyectiva?

Solución: Al utilizar la instrucción **Nucleo** en *Mathematica* se obtiene:

In[ ]:=

$T[\{x\_,y\_,z\_,w\_\}]:=\left\{\frac{1}{3}(w-13x+8y-9z),w-3x+2y-2z\right\}$

Nucleo$[T,4,\{0,0\}]$

Out[ ]=

Nucleo=Gen $\begin{pmatrix} 1 & 0 & -\frac{10}{7} & \frac{1}{7} \\ 0 & 1 & \frac{6}{7} & -\frac{2}{7} \end{pmatrix}$

No es inyectiva

Nucleo utiliza los mismos argumentos del comando **TLQ**, es decir, la aplicación lineal, la dimensión del dominio y el cero del codominio. Además, antes de realizar cualquier cálculo, verifica si T es una transformación lineal, si no lo es, se lo indica al usuario.

Ejercicio

Determine el núcleo de las siguientes funciones. En cada caso establezca su inyectividad.

(a) $T:\mathbb{R}^3\to\mathbb{R}^3$ con $T(x,y,z)=(x,y+z,x+z)$.

(b) $T:\mathbb{R}^3\to\mathbb{R}^3$ con $T(x,y,z)=(z,2x+y+z,2x+y)$.

(c) $T:\mathbb{R}^4\to\mathbb{R}^4$ con $T(x,y,z,w)=(0,2x+y-w,2x+y-w,z)$.

(d) $T:M_2(\mathbb{R})\to M_2(\mathbb{R})$ con $T(A)=A^t$.

3. **Imagen**: determina como un conjunto generado la imagen de una transformación lineal de $\mathbb{R}^n$ a $\mathbb{R}^m$. Además, indica si la aplicación es sobreyectiva. Sintaxis: **Imagen[T, n, cero]**, los argumentos que recibe son iguales a los del comando **TLQ**.

Ejemplo 1.60

Sea la aplicación lineal $T:M_2(\mathbb{R})\to\mathbb{R}^2$, tal que:

$$T\begin{pmatrix} x & y \\ z & w \end{pmatrix}=\left(\frac{-13x+8y-9z+w}{3},-3x+2y-2z+w\right)$$

Halle su imagen, ¿es T sobreyectiva?

Solución: La función Imagen resuelve lo solicitado:

In[ ]:=

$T[\{x\_,y\_,z\_,w\_\}]:=\left\{\frac{1}{3}(w-13x+8y-9z),w-3x+2y-2z\right\}$

Imagen$[T,4,\{0,0\}]$

Out[ ]=

Imagen=Gen $\begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}$

Es sobreyectiva

El lector preverá que la instrucción **Imagen**, recibe los mismos parámetros de los comandos **TLQ** y **Nucleo**. **Imagen** al igual que **Nucleo**, corrobora que la función T recibida sea una aplicación lineal, en caso contrario, envía un mensaje al usuario sin realizar ningún cálculo.

Ejercicio

Determine la imagen de las siguientes funciones. En cada caso establezca su sobreyectividad.

(a) $T:\mathbb{R}^3\to\mathbb{R}^3$ con $T(x,y,z)=(x,y+z,x+z)$.

(b) $T:\mathbb{R}^3\to\mathbb{R}^3$ con $T(x,y,z)=(z,2x+y+z,2x+y)$.

(c) $T:\mathbb{R}^4\to\mathbb{R}^4$ con $T(x,y,z,w)=(0,2x+y-w,2x+y-w,z)$.

(d) $T:M_2(\mathbb{R})\to M_2(\mathbb{R})$ con $T(A)=A^t$.

(e) $T:\mathbb{R}^4\to\mathbb{R}^4$ con $T(x,y,z,w)=(0,0,0,0)$.

4. **TL**: encuentra el criterio de una aplicación lineal dadas algunas imágenes en la lista **Imagenes_List** cuyas preimágenes forman una base del dominio contenida en la lista **PreImagenes_List**. Sintaxis: **TL[PreImagenes_List, Imagenes_List, DimensionDominio]**, **DimensionDominio**es la dimensión del espacio vectorial dominio.

Ejemplo 1.61

Halle el criterio de la transformación lineal $T:\mathbb{R}^3\to\mathbb{R}^2$, sabiendo que:

$$T(1,2,0)=(1,1)$$
$$T(-1,1,2)=(1,1)$$
$$T(0,3,3)=(-1,0)$$

Solución: **TL** retorna:

In[ ]:= TL $\left[\begin{pmatrix} 1 & 2 & 0 \\ -1 & 1 & 2 \\ 0 & 3 & 3 \end{pmatrix}, \begin{pmatrix} 1 & 1 \\ 1 & 1 \\ -1 & 0 \end{pmatrix}, 3\right]$

Out[ ]= $\left\{\frac{1}{3}(-13x_1+8x_2-9x_3), -3x_1+2x_2-2x_3\right\}$

El valor 3 que aparece como último parámetro del método **TL**, representa la dimensión del espacio vectorial dominio. **TL** verifica si las preimágenes ingresadas por el usuario, forman una base del dominio, si no es así, lo indica mediante un mensaje de texto.

Ejercicio

Encuentre el criterio de la aplicación lineal T, bajo las condiciones indicadas:

(a) $T:\mathbb{R}^2\to\mathbb{R}^3$ con $T(1,1)=(1,0,-1)$ y $T(1,-1)=(1,0,1)$.

(b) $T:\mathbb{R}^3\to\mathbb{R}^3$ con $T(1,0,0)=(2,4,-6)$, $T(0,1,0)=(-1,-2,3)$ y $T(0,0,1)=(3,6,-9)$.

(c) $T:\mathbb{R}^3\to\mathbb{R}^3$ con $T(0,0,1)=(1,1,0)$, $T(0,1,1)=(1,2,1)$ y $T(1,0,1)=(1,3,2)$.

5. **MatrizRepreTL**: determina la matriz representativa de una transformación lineal. Sintaxis: **MatrizRepreTL [T, B1, B2, DimensionDominio, DimensionCodominio, CeroCodominio]**, **T** es el criterio de la función, **B1**es una base del dominio, **B2** es una base del codominio, **DimensionDominio** y **DimensionCodominio** son las dimensiones indicadas y **CeroCodominio** es el cero del espacio vectorial codominio**.**

Ejemplo 1.62

Encuentre $[T]_{B_1}^{B_2}$ de la aplicación lineal $T:\mathbb{R}^2\to\mathbb{R}^2$ con $T(x,y)=(x-y,x+2y)$, $B_1=\{(-1,1),(0,2)\}$ y $B_2=\{(-1,0),(0,1)\}$.

Solución: En *Mathematica*:

In[ ]:=

$T[\{x\_,y\_\}]$**:=**$\{x-y,x+2y\}$

B1 $=\begin{pmatrix}-1 & 1\\ 0 & 2\end{pmatrix}$;

B2 $=\begin{pmatrix}-1 & 0\\ 0 & 1\end{pmatrix}$;

MatrizRepreTL$[T,$**B1**$,$**B2**$,2,2,\{0,0\}]$

Out[ ]= $\begin{pmatrix}2 & 2\\ 1 & 4\end{pmatrix}$

Los parámetros 2, 2 y $\{0,0\}$ corresponden a la dimensión del dominio, codominio y cero del codominio, respectivamente. Estos datos permiten validar a T como una transformación lineal, en caso de que no lo sea, **MatrizRepreTL** no halla la matriz representativa, enviando un mensaje del porque al usuario. También, **MatrizRepreTL** verifica que **B1** y **B2**

sean bases de cada espacio vectorial, si alguna no lo fuera, manda a imprimir "Error en el ingreso de las bases".

Ejercicio

Halle $[T]_{B_1}^{B_2}$ de acuerdo con:

(a) $T:\mathbb{R}^3\to\mathbb{R}^2$ con $T(x,y,z)=(4x+y,-5y+2z)$, $B_1=\{(1,0,0),(0,1,0),(0,0,1)\}$ y $B_2=\{(1,0),(0,1)\}$.

(b) $T:\mathbb{R}^4\to\mathbb{R}^4$ con $T(x,y,z,w)=(0,x,y,z)$, $B_1=\{(1,0,0,1),(-1,0,0,1),(0,1,1,0),(0,1,-1,0)\}$ y $B_2=B_1$.

(c) $T:\mathbb{R}^2\to\mathbb{R}^3$ con $T(x,y)=\left(\frac{2}{11}(x+2y),\frac{1}{11}(-29x+19y),\frac{1}{11}(-17x+21y)\right)$, $B_1=\{(2,1),(-1,2)\}$ y $B_2=\{(1,0,0),(0,1,0),(0,0,1)\}$.

6. **`TLMatrizRepre`**: encuentra el criterio de una aplicación lineal dada una matriz representativa. Sintaxis: **`TLMatrizRepre[A, B1, B2, DimensionDominio, DimensionCodominio]`**, **`A`** es la matriz representativa de la base **`B1`** a las base **`B2`** y **`DimensionDominio`**y **`DimensionCodominio`**son las dimensiones de los espacios vectoriales señalados.

Ejemplo 1.63

Determine el criterio de la transformación lineal $T:\mathbb{R}^3\to\mathbb{R}^3$ donde:

$$[T]_{B_1}^{B_2}=\begin{pmatrix}1&-1&3\\2&0&1\\3&-1&2\end{pmatrix}$$

$B_1=\{(-1,3,0),(0,2,1),(0,0,1)\}$ y B_2 la base canónica de $\mathbb{R}^3$.

Solución: **`TLMatrizRepre`** resuelve lo indicado en el ejemplo:

```
In[ ]:= TLMatrizRepre[{{{1,-1,3},{2,0,1},{3,-1,2}},{{-1,3,0},{0,2,1},{0,0,1}},{{1,0,0},{0,1,0},{0,0,1}},3,3}]
Out[ ]= {-7x1 - 2x2 + 3x3, -7x1/2 + x3 - x2/2, 1/2 (-15x1 - 3x2 + 4x3)}
```

Los dos últimos argumentos incluidos en el método **`TLMatrizRepre`**, son las dimensiones del dominio y codominio de la aplicación lineal, respectivamente. Antes de realizar los cálculos correspondientes, **`TLMatrizRepre`** corrobora el tamaño de la matriz representativa y si las dos matrices adicionales ingresadas, corresponden a bases de cada

uno de los espacios vectoriales dominio y codominio, en caso de existir una inconsistencia lo indica al usuario.

Ejercicio

Halle el criterio de la transformación lineal T, donde: $T : \mathbb{R}^2 \to \mathbb{R}^3$, $B_1 = \{(3,4),(-2,1)\}$, B_2 es la base canónica de $\mathbb{R}^3$ y:

$$[T]_{B_1}^{B_2} = \begin{pmatrix} 2 & 0 \\ -1 & 7 \\ 3 & 5 \end{pmatrix}$$

7. **MatrizPasaje**: encuentra una matriz de pasaje o de cambio de base de la base **B1** a la base **B2**. Sintaxis: **MatrizPasaje[B1, B2]**.

Ejemplo 1.64

Determine la matriz de cambio de base sobre $\mathbb{R}^2$ con $B_1 = \{(-1,1),(0,2)\}$ y $B_2 = \{(-1,0),(0,1)\}$.

Solución: El comando **MatrizPasaje** encuentra la matriz solicitada:

In[ ]:=

$$\mathbf{B1} = \begin{pmatrix} -1 & 1 \\ 0 & 2 \end{pmatrix};$$

$$\mathbf{B2} = \begin{pmatrix} -1 & 0 \\ 0 & 1 \end{pmatrix};$$

MatrizPasaje[B1,B2]

Out[ ]=

$$\begin{pmatrix} 1 & 0 \\ 1 & 2 \end{pmatrix}$$

MatrizPasaje verifica que **B1** y **B2** sean bases de un mismo espacio vectorial, de lo contrario, indica el error mediante un mensaje de texto.

Ejercicio

Halle la matriz de pasaje en cada caso:

(a) En $\mathbb{R}^3$ con $B_1 = \{(1,0,1),(0,1,1),(-1,0,1)\}$ y $B_2 = \{(0,-1,1),(1,0,-1),(1,0,1)\}$.

(b) En $\mathbb{R}^2$ con $B_1 = \{(-1,1),(0,2)\}$ y $B_2 = \{(-1,0),(0,1)\}$.

8. **TLComposicion**: Determina el criterio de la composición ToS entre dos transformaciones lineales en $\mathbb{R}^n$.

Sintaxis: **TLComposicion[S, T, DimensionEspacio1, DimensionEspacio2,DimensionEspacio3, CeroCodominio1, CeroCodominio2]**.

Ejemplo 1.65

Encuentre la aplicación lineal composición ToS, con $S : \mathbb{R}^3 \to \mathbb{R}^2$, $T : \mathbb{R}^2 \to \mathbb{R}^4$ tales que:

$$S(x,y,z) = (x+y+z, 2x+y)$$
$$T(x,y) = (x, y-x, x-y, x+y)$$

Solución: En el software:

```
In[ ]:=
 S[{x_,y_,z_}]:={x + y + z,2x + y}
 T[{x_,y_}]:={x,y - x,x - y,x + y}
 TLComposicion[S,T,3,2,4,{0,0},{0,0,0,0}]
Out[ ]= {x1 + x2 + x3,x1 - x3,x3 - x1,3x1 + 2x2 + x3}
```

Los argumentos 3, 2, 4, $\{0,0\}$, $\{0,0,0,0\}$ representan respectivamente: la dimensión del dominio de S, la dimensión del dominio de T, la dimensión del codominio de T, el cero del codominio de S y el cero del codominio de T. El comando corrobora que las funciones S y T sean transformaciones lineales.

9. **TLMatrizRepreCB**: aplica el teorema de cambio de bases, calcula **[T](B3, B4)**, usando **[T](B1, B2)**.

Sintaxis: **TLMatrizRepreCB[T, B1, B2, B3, B4,DimensionDominio, DimensionCodominio, CeroCodominio]**.

Ejemplo 1.66

Determine $[T]_{B_3}^{B4}$ a través de $[T]_{B_1}^{B_2}$ si $T : \mathbb{R}^3 \to \mathbb{R}^3$, $T(x,y,z) = (z, y-x, x-y)$, $B_1 = \{(1,0,1),(1,1,0),(0,0,1)\}$, $B_2 = B_1$, $B_3 = \{(1,0,0),(0,1,0),(0,1,1)\}$ y $B_4 = B_3$.

Solución: **TLMatrizRepreCB** ejecuta lo requerido para solucionar este ejemplo:

In[ ]:=

$T[\{x\_,y\_,z\_\}]:=\{z,y-x,x-y\}$

$\mathbf{B1}=\begin{pmatrix}1&0&1\\1&1&0\\0&0&1\end{pmatrix};$

$\mathbf{B2}=\begin{pmatrix}1&0&0\\0&1&0\\0&1&1\end{pmatrix};$

MatrizRepreTLCB$[T,$**B1**$,$**B1**$,$**B2**$,$**B2**$,3,3,\{0,0,0\}]$

Out[ ]=

$$[I]_{(B2,B4)}=\begin{pmatrix}1&1&0\\-1&1&-1\\1&0&1\end{pmatrix}\ [I]_{(B3,B1)}=\begin{pmatrix}1&-1&-1\\0&1&1\\-1&1&2\end{pmatrix}\ [T]_{(B3,B4)}=\begin{pmatrix}0&0&1\\-2&2&2\\1&-1&-1\end{pmatrix}$$

TLMatrizRepreCB verifica que T sea una transformación lineal y que cada matriz B corresponda a una base.

Ejercicio

Halle $[T]_{B_3}^{B4}$ por medio de $[T]_{B_1}^{B_2}$ si $T:\mathbb{R}^2\to\mathbb{R}^3$:

$$T(x,y)=\left(\frac{2}{11}(x+2y),\frac{1}{11}(-29x+19y),\frac{1}{11}(-17x+21y)\right)$$

$B_1=\{(3,4),(-2,1)\}$, B_2 la base canónica de $\mathbb{R}^3$, $B_3=\{(2,1),(-1,2)\}$ y $B_4=B_2$.

10. **TLInversa**: determina si una aplicación lineal es invertible y encuentra su criterio, siempre y cuando la dimensión del espacio vectorial dominio sea igual al espacio vectorial codominio. Sintaxis: **TLInversa[T, n, CeroCodominio]**, los argumentos recibidos por la función son iguales a los de la instrucción **TLQ**.

Ejemplo 1.67

Encuentre T^{-1} si existe, con $T:\mathbb{R}^4\to\mathbb{R}^4$, $T(x,y,z,w)=(x,z,y,w)$.

Solución: En *Mathematica*:

```
In[ ]:=
In[]:=
T[{x_,y_,z_,w_}]:={x,z,y,w}
TLInversa[T,4,{0,0,0,0}]
Out[ ]=
La aplicación lineal es invertible y su criterio corresponde a:
{x1,x3,x2,x4}
```

TLInversa corrobora que T sea una aplicación lineal y además, una transformación biyectiva, en caso de no cumplir alguna de estas condiciones lo indica al usuario.

Ejercicio

Determine en cada caso T^{-1} si existe.

(a) $T:\mathbb{R}^3\to\mathbb{R}^3$, $T(x,y,z)=(x,y+z,x+z)$.

(b) $T:\mathbb{R}^3\to\mathbb{R}^3$, $T(x,y,z)=(z,2x+y+z,2x+y)$.

1.9 Diagonalización de matrices y operadores con VilGebra

La diagonalización de matrices y operadores es otro tema de vital importancia en cursos básicos de álgebra lineal. **VilGebra** a este respecto, aporta algunas instrucciónes para determinar si una matriz es no diagonalizable, diagonalizar matrices y operadores lineales, resolver diagonalizaciones ortogonales, calcular la potencia $n-$ésima de una matriz a partir de su forma diagonal y diagonalización de formas cuadráticas. Las herramientas compartidas pueden ser utilizadas por los estudiantes para verificar los procedimientos resueltos con papel y lápiz.

1. **DiagonalizacionQ**: retorna **True** si una matriz cuadrada A es diagonalizable y **False** en caso contrario. Sintaxis: **DiagonalizacionQ[A]**.

Ejemplo 1.68

Dada la siguiente matriz establezca si es o no diagonalizable:

$$A = \begin{pmatrix} 3 & -2 & 0 \\ -2 & -3 & 0 \\ 0 & 0 & 5 \end{pmatrix}$$

Solución: Recurriendo al comando **DiagonalizacionQ** se obtiene:

```
In[ ]:=
A = {{3, -2, 0}, {-2, -3, 0}, {0, 0, 5}};
DiagonalizacionQ[A]
Out[ ]= True
```

Se concluye que A es diagonalizable. **DiagonalizacionQ** valida si el argumento recibido corresponde a una matriz, en caso contrario, lo indica al usuario.

Ejercicio

Mediante el uso de la instrucción **DiagonalizacionQ**, determine si cada matriz es diagonalizable:

(a) $\begin{pmatrix} 1 & 2 & 2 \\ 0 & 2 & 1 \\ -1 & 2 & 2 \end{pmatrix}$

(b) $\begin{pmatrix} 1 & 1 & 1 & 1 \\ 1 & 1 & -1 & -1 \\ 1 & -1 & 1 & -1 \\ 1 & -1 & -1 & 1 \end{pmatrix}$

(c) $\begin{pmatrix} -1 & -1 & -6 & 3 \\ 1 & -2 & -3 & 0 \\ -1 & 1 & 0 & 1 \\ -1 & -1 & 5 & 3 \end{pmatrix}$

(d) $\begin{pmatrix} -1 & -1 & -6 & 3 & 1 \\ 1 & -2 & -3 & 0 & 0 \\ -1 & 1 & 0 & 1 & 2 \\ -1 & -1 & 5 & 3 & -1 \end{pmatrix}$

2. **Diagonalizacion**: diagonaliza una matriz A diagonalizable. Sintaxis: **Diagonalizacion[A]**.

Ejemplo 1.69

Encuentre la matriz invertible P y la matriz diagonal D, que diagonaliza:

$$A = \begin{pmatrix} 3 & -2 & 0 \\ -2 & -3 & 0 \\ 0 & 0 & 5 \end{pmatrix}$$

Solución: En *Mathematica*:

```
In[ ]:=
```

$$A = \begin{pmatrix} 3 & -2 & 0 \\ -2 & -3 & 0 \\ 0 & 0 & 5 \end{pmatrix};$$

Diagonalizacion$[A]$

```
Out[ ]=
```

$$\mathbf{P=}\begin{pmatrix} 0 & \frac{1}{2}\left(-3+\sqrt{13}\right) & \frac{1}{2}\left(-3-\sqrt{13}\right) \\ 0 & 1 & 1 \\ 1 & 0 & 0 \end{pmatrix}$$

$$\mathbf{D=}\begin{pmatrix} 5 & 0 & 0 \\ 0 & -\sqrt{13} & 0 \\ 0 & 0 & \sqrt{13} \end{pmatrix}$$

El comando **Diagonalizacion** verifica que su argumento sea una matriz diagonalizable.

Ejercicio

Diagonalice si es posible la siguiente matriz:

$$A = \begin{pmatrix} 1 & 1 & 1 & 1 \\ 1 & 1 & -1 & -1 \\ 1 & -1 & 1 & -1 \\ 1 & -1 & -1 & 1 \end{pmatrix}$$

3. **DiagonalizacionOrtogonal**: diagonaliza ortogonalmente una matriz A simétrica. Sintaxis: **DiagonalizacionOrtogonal[A]**.

Ejemplo 1.70

Diagonalice ortogonalmente la matriz:

$$A = \begin{pmatrix} 1 & 2 & -4 \\ 2 & 4 & 2 \\ -4 & 2 & 1 \end{pmatrix}$$

Solución: **DiagonalizacionOrtogonal** retorna:

In[ ]:=

$A = \begin{pmatrix} 1 & 2 & -4 \\ 2 & 4 & 2 \\ -4 & 2 & 1 \end{pmatrix};$

DiagonalizacionOrtogonal$[A]$

Out[ ]=

P= $\begin{pmatrix} -\frac{1}{\sqrt{2}} & \frac{1}{3\sqrt{2}} & \frac{2}{3} \\ 0 & \frac{2\sqrt{2}}{3} & -\frac{1}{3} \\ \frac{1}{\sqrt{2}} & \frac{1}{3\sqrt{2}} & \frac{2}{3} \end{pmatrix}$

D= $\begin{pmatrix} 5 & 0 & 0 \\ 0 & 5 & 0 \\ 0 & 0 & -4 \end{pmatrix}$

La instrucción corrobora que el parámetro ingresado por el usuario sea una matriz simétrica.

Ejercicio

Diagonalice ortogonalmente si es posible, las siguientes matrices:

(a) $\begin{pmatrix} 2 & -4 & 2 \\ -4 & 2 & -2 \\ 2 & -2 & -1 \end{pmatrix}$

(b) $\begin{pmatrix} 3 & -2 & 0 \\ -2 & -3 & 0 \\ 0 & 0 & 5 \end{pmatrix}$

(c) $\begin{pmatrix} 1 & 1 & 1 & 1 \\ 1 & 1 & -1 & -1 \\ 1 & -1 & 1 & -1 \\ 1 & -1 & -1 & 1 \end{pmatrix}$

4. **PotenciaNMatriz**: calcula la potencia n-ésima de una matriz diagonalizable (similar a MatrixPower). Sintaxis: **PotenciaNMatriz[A, expon]**, lo anterior retorna $\mathbf{A}^{\text{expon}}$.

Ejemplo 1.71

Halle A^n donde:

$$A = \begin{pmatrix} 3 & -2 & 0 \\ -2 & -3 & 0 \\ 0 & 0 & 5 \end{pmatrix}$$

Solución: En *Mathematica*:

```
In[ ]:=
A = {{3, -2, 0}, {-2, -3, 0}, {0, 0, 5}};
PotenciaNMatriz[A, n]
```

```
Out[ ]=
```

$$\begin{pmatrix} \frac{1}{2}13^{\frac{n-1}{2}}\left(3-3(-1)^n+\sqrt{13}+(-1)^n\sqrt{13}\right) & 13^{\frac{n-1}{2}}\left(-1+(-1)^n\right) & 0 \\ 13^{\frac{n-1}{2}}\left(-1+(-1)^n\right) & \frac{1}{2}13^{\frac{n-1}{2}}\left(-3+3(-1)^n+\sqrt{13}+(-1)^n\sqrt{13}\right) & 0 \\ 0 & 0 & 5^n \end{pmatrix}$$

PotenciaNMatriz verifica que el argumento incluido sea una matriz y además, una matriz diagonalizable.

Ejercicio

Encuentre A^n en cada caso.

(a) $A = \begin{pmatrix} 2 & -4 & 2 \\ -4 & 2 & -2 \\ 2 & -2 & -1 \end{pmatrix}$

(b) $A = \begin{pmatrix} 1 & 1 & 1 & 1 \\ 1 & 1 & -1 & -1 \\ 1 & -1 & 1 & -1 \\ 1 & -1 & -1 & 1 \end{pmatrix}$

5. **EigensystemOL**: función Eigensystem para un operador lineal. La instrucción Eigensystem en *Mathematica* calcula en una matriz sus valores propios y vectores propios asociados. Sintaxis: **EigensystemOL[T, n, CeroCodominio]**, usa los mismos parámetros del comando **TLQ**.

Ejemplo 1.72

Halle los valores propios y vectores propios asociados, del operador lineal $T : \mathbb{R}^3 \to \mathbb{R}^3$, tal que: $T(x,y,z) = (x-z, y+z, z)$.

Solución: Usando **EigensystemOL** se tiene:

```
In[ ]:=
T[{x_,y_,z_}]:={x − z,y + z,z}
EigensystemOL[T,3,{0,0,0}]
```

```
Out[ ]=
```

Matriz representativa del operador lineal en la base can'onica: $\begin{pmatrix} 1 & 0 & -1 \\ 0 & 1 & 1 \\ 0 & 0 & 1 \end{pmatrix}$

Valores y vectores propios del operador lineal:

$$\begin{pmatrix} 1 & 1 & 1 \\ \{0,1,0\} & \{1,0,0\} & \{0,0,0\} \end{pmatrix}$$

EigensystemOL corrobora que T sea una transformación lineal y ella a su vez, un operador lineal. Los argumentos 3 y $\{0,0,0\}$ corresponden a la dimensión del dominio y al cero del codominio, respectivamente.

Ejercicio

Emplee el comando **EigensystemOL** sobre las siguientes funciones:

(a) $T : \mathbb{R}^3 \to \mathbb{R}^3$, tal que: $T(x,y,z) = (5x+4y+z, 4x+8y-4z, x-4y+5z)$.

(b) $T : \mathbb{R}^4 \to \mathbb{R}^4$, tal que: $T(x,y,z,w) = (0,x,y,z)$.

(c) $T : \mathbb{R}^3 \to \mathbb{R}^3$, tal que: $T(x,y,z) = (x+y+z, x-y+z, x+y-z)$.

6. **DiagonalizacionFC2D**: diagonaliza una forma cuadrática en el plano.

Sintaxis: **DiagonalizacionFC2D[Ecuacion_List, a]**, **Ecuacion_List** recibe en un vector la ecuación de la sección cónica y "**a**" determina el intervalo de graficación **[-a, a]**.

Ejemplo 1.73

Diagonalice la forma cuadrática:

$$x_1^2 - 4x_2x_1 + x_2^2 - 4 = 0$$

Solución: En el software:

```
In[ ]:=
DiagonalizacionFC2D[{x1^2 - 4x2x1 + x2^2 - 4 == 0}, 10]

Out[ ]=
```

La matriz de la forma cuadrática es: $\begin{pmatrix} 1 & -2 \\ -2 & 1 \end{pmatrix}$

Las matrices de transformación son:
$\begin{pmatrix} \frac{\mathbf{x}_1}{\sqrt{2}} + \frac{\mathbf{x}_2}{\sqrt{2}} \\ \frac{\mathbf{x}_2}{\sqrt{2}} - \frac{\mathbf{x}_1}{\sqrt{2}} \end{pmatrix} \begin{pmatrix} \frac{\mathbf{y}_1}{\sqrt{2}} - \frac{\mathbf{y}_2}{\sqrt{2}} \\ \frac{\mathbf{y}_1}{\sqrt{2}} + \frac{\mathbf{y}_2}{\sqrt{2}} \end{pmatrix}$

La ecuación original se reduce a:
$y_1^2 + 4 = 3y_2^2$
El ángulo de rotación es: $\frac{\pi}{4}$

La gráfica corresponde a:

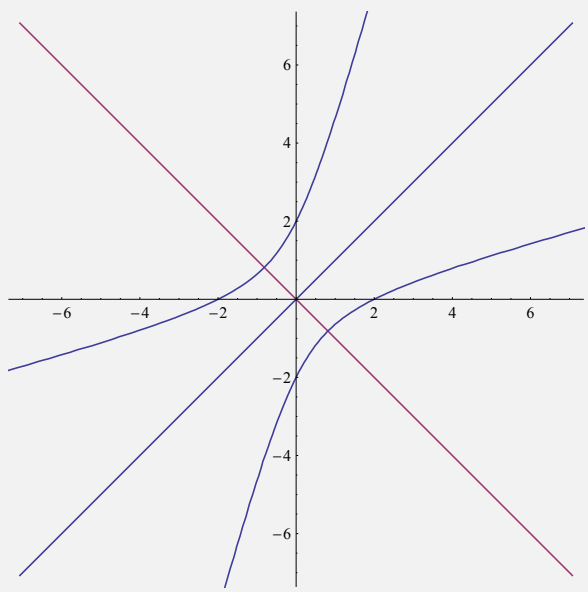

Figura 20: Sección cónica con ejes rotados

El 10 recibido como parámetro del método establece un intervalo de graficación en los ejes coordenados.

Ejercicio

Diagonalice en $\mathbb{R}^2$ la forma cuadrática:

$$5x_1^2 + 6x_2x_1 - 4x_1 + 5x_2^2 + 4x_2 = 4$$

7. **`DiagonalizacionFC3D`**: diagonaliza una forma cuadrática en el espacio.

Sintaxis: **`DiagonalizacionFC3D[Ecuacion_List, a]`**, **`Ecuacion_List`** recibe en un vector la ecuación de la superficie cuadrática y "**`a`**" determina el intervalo de graficación **`[-a, a]`**.

Ejemplo 1.74

Diagonalice la forma cuadrática:

$$144x_1^2 - 216x_3x_1 - 540x_1 + 100x_2^2 + 81x_3^2 - 720x_3 = 0$$

Solución: **`DiagonalizacionFC3D`** retorna:

```
In[ ]:=
DiagonalizacionFC3D[{144x1^2 - 216x3x1 - 540x1 + 100x2^2 + 81x3^2 - 720x3 == 0}, 10, 0.5]

Out[ ]=
```

La matriz de la forma cuadrática es: $\begin{pmatrix} 144 & 0 & -108 \\ 0 & 100 & 0 \\ -108 & 0 & 81 \end{pmatrix}$

Las matrices de transformación son:

$$\begin{pmatrix} \mathbf{x}_2 \\ \frac{3\mathbf{x}_3}{5} - \frac{4\mathbf{x}_1}{5} \\ \frac{3\mathbf{x}_1}{5} + \frac{4\mathbf{x}_3}{5} \end{pmatrix} \begin{pmatrix} \frac{\mathbf{y}_3}{5} - \frac{4\mathbf{y}_2}{5} \\ \mathbf{y}_1 \\ \frac{3\mathbf{y}_2}{5} + \frac{4\mathbf{y}_3}{5} \end{pmatrix}$$

La ecuación original se reduce a:

$4y_1^2 + 9y_2^2 = 36y_3$

Se aclara al lector que el valor 0.5 incluido como argumento de la función **`DiagonalizacionFC3D`**, define un porcentaje de opacidad sobre la gráfica de la superficie cuadrática correspondiente. En este ejemplo, 0.5 es equivalente a un 50% de transparencia.

La gráfica corresponde a:

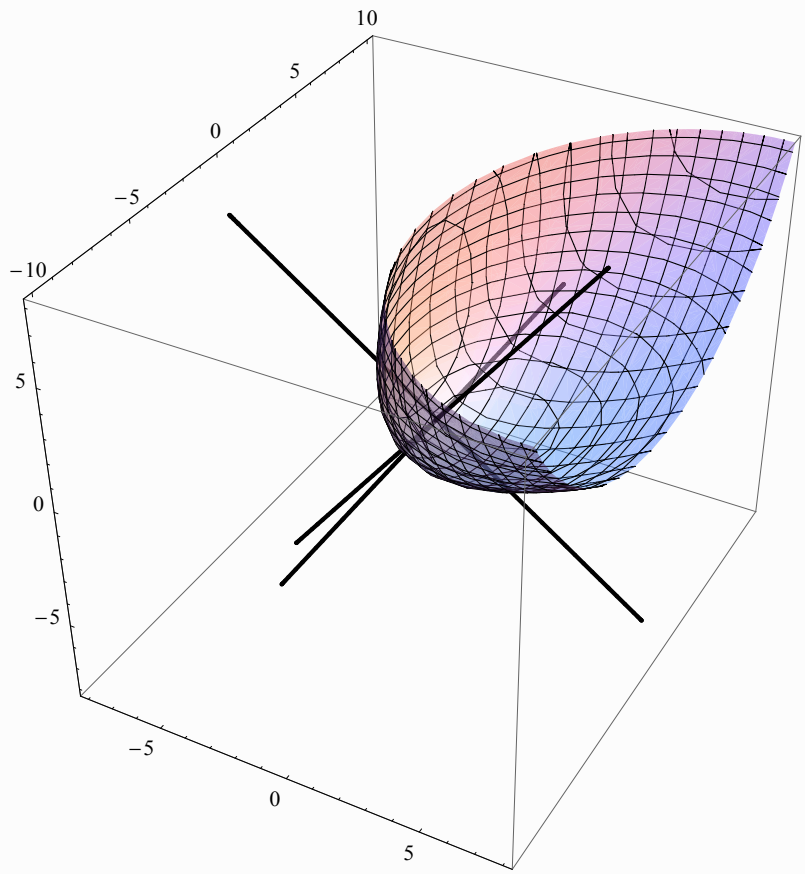

Figura 21: Superficie cuadrática con ejes rotados

Ejercicio

Diagonalice en $\mathbb{R}^3$ la forma cuadrática:

$$-4x_1^2 + 4x_2x_1 + 10x_3x_1 - 4x_2^2 - 7x_3^2 + 10x_2x_3 = 3$$

1.10 Programación lineal con `VilGebra`

La programación lineal ofrece algoritmos que demandan muchas veces cálculos significativos por la cantidad de operaciones que involucran. La experiencia docente del autor ha demostrado cómo los métodos clásicos de resolución (el gráfico y el símplex) en ocasiones resultan complejos a las poblaciones estudiantiles, no por los algoritmos en sí mismos, sino por sus factores operativos. En este sentido, el uso de software puede contribuir ha alivianar la carga aritmética o algebraica permitiéndo

Paquete ***VilGebra*** . Enrique Vílchez Q.

al docente concentrarse en los elementos esenciales de su enseñanza, relacionados con el análisis y la interpretación de resultados. Todos los comandos expuestos a continuación presentan dicho enfoque, intentando brindar opciones de cálculo que faciliten espacios de comprensión y resolución de problemas.

1. **PLMG1**: resuelve un problema de programación lineal utilizando el método gráfico siempre y cuando la región factible sea acotada. Sintaxis: **PLMG1[F, o, Restricciones, vgx, vgy]**, **F** es la función objetivo, **o** debe ser 1 si es un problema de máximos o 0 si es de mínimos, **Restricciones** representa la lista de desigualdades del problema, sin considerar las condiciones de no negatividad, **vgx** es un valor de graficación sobre el eje x y **vgy** sobre el eje y.

2. **PLMG2**: resuelve un problema de programación lineal utilizando el método gráfico siempre y cuando la región factible sea acotada. Sintaxis: **PLM2[MFO, o, MCN, MTC, vgx, vgy]**, **MFO** es un vector que contiene los coeficientes numéricos de la función objetivo, **o** debe ser 1 si es un problema de máximos o 0 si es de mínimos, **MCN** representa una matriz de coeficientes numéricos del sistema de desigualdades del problema, sin considerar las condiciones de no negatividad, **MTC** es una matriz de términos constantes que incluye una segunda columna donde se digita 1 si la restricción es $\geq$ o -1 si es $\leq$, **vgx** es un valor de graficación sobre el eje x y **vgy** sobre el eje y.

Ejemplo 1.75

Maximice $Z = 2x_1 + x_2$ sujeta a:

$$\begin{cases} x_2 \leq 10 \\ 2x_1 + 5x_2 \leq 60 \\ x_1 + x_2 \leq 18 \\ 3x_1 + x_2 \leq 44 \\ x_1 \geq 0, x_2 \geq 0 \end{cases}$$

Solución: Usando **PLMG1** se obtiene:

```
In[ ]:=
F[x_,y_]:=2x + y
PLMG1[F,1,{y ≤ 10,2x + 5y ≤ 60,x + y ≤ 18,3x + y ≤ 44},15,10.5]
```

Out[ ]=

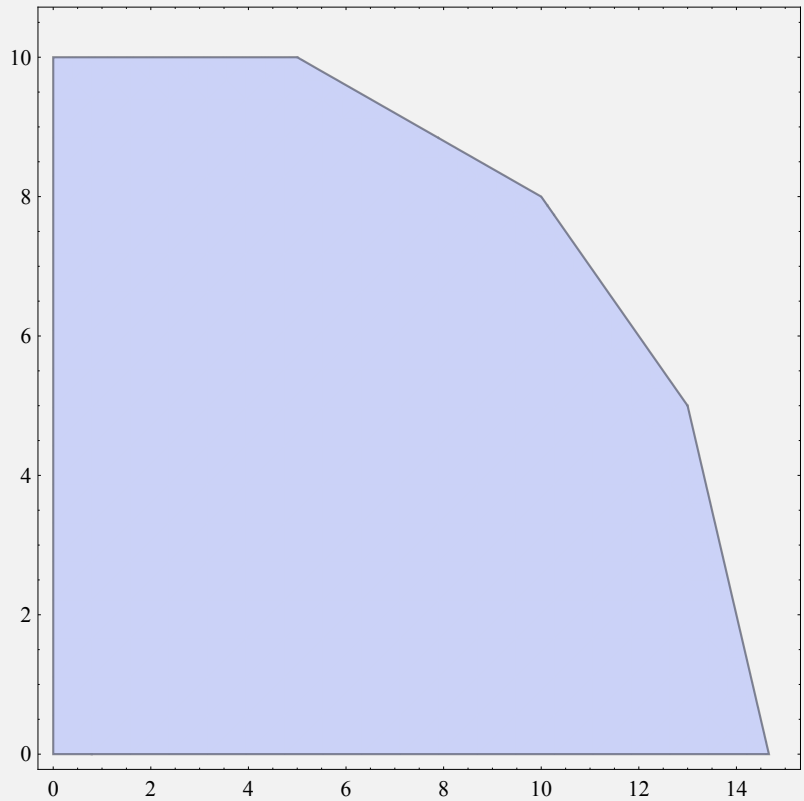

Figura 22: Región factible

Los vértices son: $\begin{pmatrix} 0 & 0 \\ 0 & 10 \\ 5 & 10 \\ 10 & 8 \\ 13 & 5 \\ \frac{44}{3} & 0 \end{pmatrix}$

Sus imágenes son respectivamente: $\{0,10,20,28,31,\frac{88}{3}\}$

Hay una única solución que ocurre en: $\{13,5\}$ **y corresponde a: 31**

PLMG2 a diferencia de **PLMG1**, recibe todos los datos como arreglos unidimensionales o bidimensionales:

In[ ]:= PLMG2 $\left[\{2,1\},1,\begin{pmatrix} 0 & 1 \\ 2 & 5 \\ 1 & 1 \\ 3 & 1 \end{pmatrix},\begin{pmatrix} 10 & -1 \\ 60 & -1 \\ 18 & -1 \\ 44 & -1 \end{pmatrix},15,\mathbf{10.5}\right]$

Ambos métodos producen la misma salida. El segundo parámetro de los dos comandos indica si el problema es de máximos (digitando un 1), o bien, si es de mínimos (ingresando un 0). Los dos últimos valores definen un intervalo de graficación para la región factible en el eje x y el eje y, respectivamente.

Ejercicio

Minimice $Z = 5x_1 + 7x_2$ sujeta a:

$$\begin{cases} 2x_1 + 3x_2 \geq 147 \\ 3x_1 + 4x_2 \geq 210 \\ x_1 + x_2 \geq 63 \\ x_1 \geq 0, x_2 \geq 0 \end{cases}$$

3. **OptiFunc1**: optimiza una función lineal recurriendo al comando **LinearProgramming** de *Mathematica*. Sintaxis: **OptiFunc1[MFOO, o, MCN, MTC]**, **MFOO** es un vector que contiene los coeficientes numéricos de la función objetivo, **o** debe ser 1 si es un problema de máximos o 0 si es de mínimos, **MCN** representa una matriz de coeficientes numéricos del sistema de desigualdades del problema, sin considerar las condiciones de no negatividad y **MTC** es una matriz de términos constantes que incluye una segunda columna donde se digita 1 si la restricción es $\geq$, 0 si es $=$ o -1 si es $\leq$.

4. **OptiFunc2**: optimiza una función lineal recurriendo a los comandos **Maximize** y **Minimize** de *Mathematica*. Sintaxis: **OptiFunc2[MFO, o, MCN, MTC]**, **MFO** es un vector que contiene los coeficientes numéricos de la función objetivo, **o** debe ser 1 si es un problema de máximos o 0 si es de mínimos, **MCN** representa una matriz de coeficientes numéricos del sistema de desigualdades del problema, sin considerar las condiciones de no negatividad y **MTC** es una matriz de términos constantes que incluye una segunda columna donde se digita 1 si la restricción es $\geq$ o -1 si es $\leq$.

Ejemplo 1.76

Maximice $Z = 4x_1 - x_2 - x_3$ sujeta a:

$$\begin{cases} 3x_1 + x_2 - x_3 \leq 4 \\ x_1 + x_2 + x_3 \leq 2 \\ x_1 \geq 0, x_2 \geq 0, x_3 \geq 0 \end{cases}$$

Solución: Al emplear **OptiFunc1**:

In[ ]:= OptiFunc1 $\left[\{4,-1,-1\}, 1, \begin{pmatrix} 3 & 1 & -1 \\ 1 & 1 & 1 \end{pmatrix}, \begin{pmatrix} 4 & -1 \\ 2 & -1 \end{pmatrix}\right]$

Out[ ]= El valor óptimo es: $\frac{11}{2}$ **y ocurre en:** $\left\{\frac{3}{2}, 0, \frac{1}{2}\right\}$

Es fundamental aclarar al lector, que el comando **OptiFunc2** recibe los mismos parámetros de **OptiFunc1**, produciendo también, la misma salida. La diferencia entre ambos es el algoritmo interno empleado por *Mathematica*. En **OptiFunc1** se utiliza la función **LinearProgramming**, mientras que en **OptiFunc2** la solución se encuentra a través de los métodos **Maximize** y **Minimize**. La segunda columna de la última matriz que entra como parámetro de **OptiFunc1** y **OptiFunc2**, especifica el tipo de desigualdad en el sistema de restricciones, un 1 indica un $\geq$ y un -1 un $\leq$.

Ejercicio

Resuelva en cada caso el problema de programación lineal propuesto. Se sugiere emplear las instrucciones **`OptiFunc1`** o **`OptiFunc2`**.

(a) Minimice $Z = 5x_1 + 7x_2$ sujeta a:

$$\begin{cases} 2x_1 + 3x_2 \geq 147 \\ 3x_1 + 4x_2 \geq 210 \\ x_1 + x_2 \geq 63 \\ x_1 \geq 0, x_2 \geq 0 \end{cases}$$

(b) Maximice $Z = x_1 + 3x_2 - 2x_3$ sujeta a:

$$\begin{cases} -x_1 - 2x_2 - 2x_3 = -6 \\ -x_1 - x_2 + x_3 \leq -2 \\ x_1 \geq 0, x_2 \geq 0, x_3 \geq 0 \end{cases}$$

5. **`SimplexVH`**: aplica el método símplex para maximizar o minimizar una función lineal donde no existen variables artificiales (solo de holgura) y soluciones degeneradas. No resuelve problemas con soluciones múltiples. Sintaxis: **`SimplexVH[MFOO, o, MCN, MTC, vco]`**, **`MFOO`** es un vector que contiene los coeficientes numéricos de la función objetivo, **`o`** debe ser 1 si es un problema de máximos o 0 si es de mínimos, **`MCN`** representa una matriz de coeficientes numéricos del sistema de desigualdades del problema, sin considerar las condiciones de no negatividad, **`MTC`** es una matriz de términos constantes y **`vco`** es una cota superior de cualquier cociente mínimo en el proceso.

Ejemplo 1.77

Maximice $Z = 3x_1 + 4x_2 + 5x_3$ sujeta a:

$$\begin{cases} 3x_1 + x_2 + 5x_3 \leq 150 \\ x_1 + 4x_2 + x_3 \leq 120 \\ 2x_1 + 2x_3 \leq 105 \\ x_1 \geq 0, x_2 \geq 0, x_3 \geq 0 \end{cases}$$

Solución: **`SimplexVH`** retorna:

```
In[ ]:=
SimplexVH[{3,4,5},1,{{3,1,5},{1,4,1},{2,0,2}},{150,120,105},1000]
```

Out[ ]=

Tabla símplex 1

$$\begin{pmatrix} 3 & 1 & 5 & 1 & 0 & 0 & 0 & 150 \\ 1 & 4 & 1 & 0 & 1 & 0 & 0 & 120 \\ 2 & 0 & 2 & 0 & 0 & 1 & 0 & 105 \\ -3 & -4 & -5 & 0 & 0 & 0 & 1 & 0 \end{pmatrix}$$

$\vdots$

Tabla símplex 4

$$\begin{pmatrix} \frac{11}{19} & 0 & 1 & \frac{4}{19} & -\frac{1}{19} & 0 & 0 & \frac{480}{19} \\ \frac{2}{19} & 1 & 0 & -\frac{1}{19} & \frac{5}{19} & 0 & 0 & \frac{450}{19} \\ \frac{16}{19} & 0 & 0 & -\frac{8}{19} & \frac{2}{19} & 1 & 0 & \frac{1035}{19} \\ \frac{6}{19} & 0 & 0 & \frac{16}{19} & \frac{15}{19} & 0 & 1 & \frac{4200}{19} \end{pmatrix}$$

El valor máximo es: $\frac{4200}{19}$

Variable1 $\rightarrow 0$

Variable2 $\rightarrow \frac{450}{19}$

Variable3 $\rightarrow \frac{480}{19}$

Las siglas **VH** de esta instrucción significan "variables de holgura", haciendo referencia al tipo de problema que resuelve. El valor 1000 que entra como último parámetro de **SimplexVH**, representa una cota superior de cualquier cociente mínimo en el proceso.

Ejercicio

Maximice $Z = 8x_1 + 2x_2$ sujeta a:

$$\begin{cases} x_1 - x_2 \leq 1 \\ x_1 + 2x_2 \leq 8 \\ x_1 + x_2 \leq 5 \\ x_1 \geq 0, x_2 \geq 0 \end{cases}$$

6. **SimplexVA**: aplica el método símplex para maximizar o minimizar una función lineal donde existen variables artificiales, sin soluciones degeneradas. No resuelve problemas con soluciones múltiples. Sintaxis: **SimplexVA[MFOO, o, MCNO, MTC, vco, M]**, **MFOO** es un vector que contiene los coeficientes numéricos de la función objetivo, **o** debe ser 1 si es un problema de máximos o 0 si es de mínimos, **MCNO** representa una matriz de coeficientes numéricos del sistema de desigualdades del problema,sin considerar las condiciones de no negatividad, **MTC** es una matriz de términos constantes, **vco** es una cota superior de cualquier cociente mínimo en el proceso y **M** es un valor positivo muy grande (se sugiere el uso de 10000 para casos pequeños). La matriz **MTC** debe quedar con valores no negativos.

Ejemplo 1.78

Maximice $Z = x_1 + 3x_2 - 2x_3$ sujeta a:

$$\begin{cases} x_1 + 2x_2 + 2x_3 = 6 \\ x_1 + x_2 - x_3 \geq 2 \\ x_1 \geq 0, x_2 \geq 0, x_3 \geq 0 \end{cases}$$

Solución: Al utilizar **SimplexVA**:

In[ ]:=

$$\textbf{SimplexVA}\left[\{1,3,-2\},1,\begin{pmatrix} 1 & 2 & 2 \\ 1 & 1 & -1 \end{pmatrix},\begin{pmatrix} 6 & 0 \\ 2 & 1 \end{pmatrix},1000,10000\right]$$

Out[ ]=

Tabla símplex previa:

$$\begin{pmatrix} 1 & 2 & 2 & 0 & 1 & 0 & 0 & 6 \\ 1 & 1 & -1 & -1 & 0 & 1 & 0 & 2 \\ -1 & -3 & 2 & 0 & 10000 & 10000 & 1 & 0 \end{pmatrix}$$

$\vdots$

Tabla símplex 5

$$\begin{pmatrix} -\frac{1}{2} & 0 & 2 & 1 & \frac{1}{2} & -1 & 0 & 1 \\ \frac{1}{2} & 1 & 1 & 0 & \frac{1}{2} & 0 & 0 & 3 \\ \frac{1}{2} & 0 & 5 & 0 & \frac{20003}{2} & 10000 & 1 & 9 \end{pmatrix}$$

El valor máximo es: 9

Variable1 $\to$ 0

Variable2 $\to$ 3

Variable3 $\to$ 0

Las siglas **VA** de este comando significan "variables artificiales". El valor 10000 que entra como último parámetro de **SimplexVA**, representa es un valor positivo muy grande (se sugiere el uso de 10000 para casos pequeños). Se advierte que la primera columna de la última matriz recibida por el método, debe quedar siempre con valores no negativos. En la segunda columna de esta misma matriz, un 1 indica un $\geq$, un 0 un $=$ y un -1 un $\leq$.

Ejercicio

Resuelva en cada caso.

(a) Maximice $Z = 2x_1 + x_2$ sujeta a:

$$\begin{cases} -x_1 + x_2 \geq 2 \\ x_1 + x_2 \leq 1 \\ x_1 \geq 0, x_2 \geq 0 \end{cases}$$

(b) Maximice $Z = 2x_1 + x_2$ sujeta a:

$$\begin{cases} x_1 + x_2 \leq 12 \\ x_1 + x_2 \leq 20 \\ -x_1 + x_2 \geq 2 \\ x_1 \geq 0, x_2 \geq 0 \end{cases}$$

(c) Minimice $Z = x_1 + 2x_2$ sujeta a:

$$\begin{cases} -2x_1 + x_2 \geq 1 \\ -x_1 + x_2 \geq 2 \\ x_1 \geq 0, x_2 \geq 0 \end{cases}$$

7. **Dual**: retorna el dual del problema primal ingresado. Sintaxis: **Dual[MFO, o, MCN, MTC]**, **MFO** es un vector que contiene los coeficientes numéricos de la función objetivo, **o** debe ser 1 si es un problema de máximos o 0 si es de mínimos, **MCN** representa una matriz de coeficientes numéricos del sistema de desigualdades, sin considerar las condiciones de no negatividad y **MTC** es una matriz de términos constantes.

Ejemplo 1.79

Halle el problema dual de:

$$\textbf{Maximice} Z = 3x_1 + 4x_2 + 5x_3 \textbf{ sujeta a:}$$
$$\begin{cases} 3x_1 + x_2 + 5x_3 \geq 150 \\ x_1 + 4x_2 + x_3 \leq 120 \\ 2x_1 + 2x_3 \geq 105 \\ x_1 \geq 0, x_2 \geq 0, x_3 \geq 0 \end{cases}$$

Solución: **Dual** produce como salida:

In[ ]:=

$$\textbf{Dual}\left[\{3,4,5\}, 1, \begin{pmatrix} 3 & 1 & 5 \\ 1 & 4 & 1 \\ 2 & 0 & 2 \end{pmatrix}, \begin{pmatrix} 150 & 1 \\ 120 & -1 \\ 105 & 1 \end{pmatrix}\right]$$

Out[ ]=

El dual del problema primal
Minimizar la función objetivo con coeficientes numéricos: $\{150, 120, 105\}$
La matriz de coeficientes numéricos del sistema de restricciones es:

$$\begin{pmatrix} 3 & 1 & 2 \\ 1 & 4 & 0 \\ 5 & 1 & 2 \end{pmatrix}$$

La matriz de términos constantes y la forma de las desigualdades
es (en la segunda columna un 1 representa $>=$ **y un -1** $<=$ **):**

$$\begin{pmatrix} 3 & -1 \\ 4 & 1 \\ 5 & -1 \end{pmatrix}$$

Ejercicio

Determine usando el comando Dual, el dual de:

$$\textbf{Maximice} Z = 2x_1 + 4x_2 \textbf{ sujeta a:}$$
$$\begin{cases} x_1 - 4x_2 \leq -8 \\ x_1 + 2x_2 \leq 16 \\ x_1 \geq 0, x_2 \geq 0 \end{cases}$$

1.11 Conclusiones

VilGebra representa un esfuerzo de implementación asistida por computadora para impartir un curso de álgebra lineal en el campo de la ingeniería. Facilita al estudiante herramientas de exploración sin necesidad de profundizar las funciones de programación incluidas en el software *Mathematica*. Lo anterior, a juicio del autor en el marco de su experiencia docente, resulta ser una ventaja que favorece el aprendizaje y los procesos de enseñanza, inmersos usualmente en el breve período de un semestre. **VilGebra** a este respecto puede contribuir con la resolución de problemas contextualizados, donde el alumno se implique en tareas de análsis e interpretación, dejando de lado los aspectos rutinarios en la resolución, a través del uso de software. La mediación docente y la adecuada selección de ejemplos apoyados con ordenador, definirán el buen uso de esta herramienta didáctica.

Se espera que mediante la divulgación del paquete **VilGebra**, éste pueda ser aprovechado por otros colegas vinculados con esta área de conocimiento y resulte ser un soporte para desarrollar la docencia con una metodología complementaria de carácter no tradicional.

Bibliografía

[1] Arce, C, Castillo, W. y González, J. (2004). Álgebra lineal. San José: Editorial de la Universidad de Costa Rica.

[2] Kolman, B. (1997). Álgebra lineal con aplicaciones y *Matlab*. México: Editorial Pearson.

[3] Shiskowski, K. y Frinkle, K. (2011). Principles of Linear Algebra with *Mathematica*. USA: Editorial Wiley.

[4] Torres, C. (2006). Enseñanza de la trigonometría asistida con calculadoras graficadoras. En Ricardo Cantoral Uriza, Olda Covián Chávez et al. (Comps.), Investigaciones sobre enseñanza y aprendizaje de las matemáticas (pp. 753-778). México: Ediciones Díaz de Santos, S.A.

[5] Vílchez, E. (2012). Álgebra lineal apoyada con *Mathematica*. Costa Rica: Editorial Tecnológica.

[6] Vílchez, E. y Monge, J. (2001) Tesis: Aplicación e Interpretación de los Valores Propios en Matemática e Ingeniería. Universidad Nacional.