y si esto de conjugar nos parece enrevesado, sería equivalente

```
% Productos escalares:
v2'*v1
v3'*v1
v1'*v3
v3'*v2
```

Hay un comando `norm` que funciona correctamente con números complejos. Por ejemplo, `norm(v1)` mostraría una aproximación de $\sqrt{7}$.

En `sagemath` existe una instrucción `dot_product` que se limita a calcular $\sum x_k y_k$ y por tanto funciona en $\mathbb{R}^n$. Para que sirva también en $\mathbb{C}^n$ hay que conjugar el segundo argumento con `conjugate`. Así el análogo del código anterior sería:

```
# Definición de los vectores:
v1 = vector(CC, [2+i, 1+i])
v2 = vector(CC, [-1+i, 2-i])
v3 = vector(CC, [3, i])

# Productos escalares:
print( v1.dot_product(v2.conjugate()) )
print( v1.dot_product(v3.conjugate()) )
print( v3.dot_product(v1.conjugate()) )
print( v2.dot_product(v3.conjugate()) )
```

Hay también un comando `norm` como en `matlab/octave`. El código anterior produce resultados con decimales. La razón para ello es que estamos trabajando en $\mathbb{C}$, indicado mediante `CC`. Una forma de conseguir resultados exactos forzando el cálculo simbólico es omitir `CC` (y la coma) o sustituirlo por `SR`. Otra forma más artificiosa es cambiar al cuerpo $K = \{a + bi \,:\, a, b \in \mathbb{Q}\}$ reemplazando las cuatro primeras líneas por

```
# Definición del nuevo cuerpo (redefine i)
K.<i> = QuadraticField(-1)
# Definición de los vectores:
v1 = vector(K, [2+i, 1+i])
v2 = vector(K, [-1+i, 2-i])
v3 = vector(K, [3, i])
```

4.2. Ortogonalidad

En el teorema de Pitágoras (Proposición 4.1.3) apareció la condición $\langle \vec{x}, \vec{y} \rangle = 0$ que en $\mathbb{R}^2$ o en $\mathbb{R}^3$ no es otra cosa que la perpendicularidad. Dentro de la jerga del álgebra lineal se prefiere el término *ortogonalidad*. Es decir, se dice que dos vectores $\vec{x}$ e $\vec{y}$ son *ortogonales* si $\langle \vec{x}, \vec{y} \rangle = 0$. Si además $\vec{x}$ e $\vec{y}$ son *unitarios* se dice que son *ortonormales*. Por extensión, diremos que una base $\mathcal{B} = \{\vec{u}_1, \dots, \vec{u}_n\}$ de un espacio vectorial con producto escalar es *base ortogonal* si los $\vec{u}_j$ son ortogonales dos a dos y que es *base ortonormal* si son ortonormales. Nada impide hablar en general de conjuntos finitos ortogonales u ortonormales aunque, como veremos en breve, excluyendo vectores nulos, estos conjuntos son base del espacio que generan, gracias a que la independencia lineal está asegurada.

Por ejemplo $\mathcal{B} = \{\vec{u}_1 = (6,4)^t,\ \vec{u}_2 = (2,-3)^t\}$ es base ortogonal de $\mathbb{R}^2$. Es base porque está compuesta por $2 = \dim \mathbb{R}^2$ vectores linealmente independientes de $\mathbb{R}^2$

y es ortogonal porque $\vec{u}_1 \cdot \vec{u}_2 = 0$. No es ortonormal porque $\|\vec{u}_1\| = 2\sqrt{13} \neq 1$ y tampoco $\vec{u}_2$ es unitario.

En $\mathbb{R}^3$,

$$\mathcal{B} = \Big\{\vec{u}_1 = \frac{1}{9}(8,4,1)^t,\ \vec{u}_2 = \frac{1}{9}(4,-7,-4)^t,\ \vec{u}_3 = \frac{1}{9}(1,-4,8)^t\Big\},$$

es base ortonormal. Son tres vectores linealmente independientes de $\mathbb{R}^3$ y la ortonormalidad se sigue comprobando $\vec{u}_1\cdot\vec{u}_2 = \vec{u}_1\cdot\vec{u}_3 = \vec{u}_2\cdot\vec{u}_3 = 0$ y $\|\vec{u}_1\| = \|\vec{u}_2\| = \|\vec{u}_3\| = 1$.

Un último ejemplo con números complejos cuyos cálculos, ya sea completos o parciales, se dejan al que quiera practicar, es:

$$\mathcal{B} = \Big\{\vec{u}_1 = (4+3i, 4, -2+6i)^t, \vec{u}_2 = (4, 4-3i, -2-6i)^t, \vec{u}_3 = (-2+6i, -2-6i, 1)^t\Big\},$$

que constituye una base ortogonal de $\mathbb{C}^3$. Se cumple $\|\vec{u}_1\| = \|\vec{u}_2\| = \|\vec{u}_3\| = 9$, por ejemplo $\|\vec{u}_1\|^2 = |4+3i|^2 + 4^2 + |-2+6i|^2 = 25+16+40 = 81$. Por tanto al dividir los vectores de $\mathcal{B}$ por 9 se obtiene una base ortonormal.

En los ejemplos anteriores la comprobación de la independencia lineal es en realidad innecesaria porque se deduce de la ortogonalidad. A pesar de que este es un resultado muy fácil es suficientemente notable para destacarlo y completarlo con el cálculo de las coordenadas. Es conveniente dar un vistazo a la demostración para convencerse de que no hay nada que memorizar.

Proposición 4.2.1. *Si $\vec{u}_1, \dots, \vec{u}_n$ son vectores ortogonales no nulos en un espacio vectorial sobre $\mathbb{R}$ o $\mathbb{C}$ con un producto escalar, entonces son linealmente independientes. Además si V es el subespacio generado por ellos, para cada $\vec{x} \in V$ se tiene*

$$\vec{x} = x_1\vec{u}_1 + x_2\vec{u}_2 + \cdots + x_n\vec{u}_n \qquad \text{con} \quad x_j = \frac{\langle \vec{x}, \vec{u}_j\rangle}{\|\vec{u}_j\|^2}.$$

Es decir, $\|\vec{u}_j\|^{-2}\langle \vec{x}, \vec{u}_j\rangle$ son las coordenadas de $\vec{x}$ en la base $\mathcal{B} = \{\vec{u}_1, \dots, \vec{u}_n\}$.

Demostración. Al calcular el producto escalar de $\vec{x} = x_1\vec{u}_1 + x_2\vec{u}_2 + \cdots + x_n\vec{u}_n$ con $\vec{u}_j$ se obtiene $\langle \vec{x}, \vec{u}_j\rangle = x_j\langle \vec{u}_j, \vec{u}_j\rangle$. Si $\vec{x} = \vec{0}$ se deduce $x_j = 0$ y con ello la independencia lineal y para $\vec{x}$ general la fórmula indicada. □

Un ejemplo de que esto funciona es hallar las coordenadas de $\vec{v} = (0,2,1)^t$ en la base ortonormal de $\mathbb{R}^3$ considerada antes. En vez de calcular x_1, x_2 y x_3 resolviendo el sistema $\vec{v} = x_1\vec{u}_1 + x_2\vec{u}_2 + x_3\vec{u}_3$, en virtud del resultado anterior, tenemos directamente

$$x_1 = \vec{v}\cdot\vec{u}_1 = 1, \qquad x_2 = \vec{v}\cdot\vec{u}_2 = -2, \qquad x_3 = \vec{v}\cdot\vec{u}_3 = 0.$$

Es fácil comprobar $\vec{v} = \vec{u}_1 - 2\vec{u}_2$ y por tanto este cálculo es correcto.

Otro ejemplo, esta vez con una base ortogonal, es el cálculo de las coordenadas de $\vec{x} = (1, 3, 7)^t$ en $\mathcal{B} = \{\vec{u}_1, \vec{u}_2, \vec{u}_3\}$ con

$$\vec{u}_1 = (2, 2, 1)^t, \qquad \vec{u}_2 = (2, -1, -2)^t, \qquad \vec{u}_3 = (1, -2, 2)^t$$

que es base ortogonal de $\mathbb{R}^3$. Se tiene $\|\vec{u}_j\| = 3$ y las fórmulas de la Proposición 4.2.1 dan

$$x_1 = \frac{15}{9} = \frac{5}{3}, \qquad x_2 = -\frac{15}{9} = -\frac{5}{3}, \qquad x_3 = \frac{9}{9} = 1.$$

El resultado se comprueba fácilmente con $\vec{x} = 5(\vec{u}_1 - \vec{u}_2)/3 + \vec{u}_3$.

Veamos un último ejemplo con números complejos hallando las coordenadas de $\vec{x} = (-5+i, 6+i)^t$ en la base ortogonal $\mathcal{B} = \{(1+i, i)^t, (-2+i, 3+i)^t\}$. Las cuentas, con detalle, son:

$$\begin{cases} x_1 = \dfrac{(-5+i)(1-i) + (6+i)(-i)}{|1+i|^2 + |i|^2} = \dfrac{-3}{3} = -1, \\[2ex] x_2 = \dfrac{(-5+i)(-2-i) + (6+i)(3-i)}{|-2+i|^2 + |3+i|^2} = \dfrac{30}{15} = 2, \end{cases}$$

que es coherente con $\vec{x} = -(1+i, i)^t + 2(-2+i, 3+i)^t$.

La moraleja es que hallar coordenadas en bases ortogonales es mucho más fácil que en el caso general: en vez de resolver sistemas de ecuaciones debemos calcular productos escalares, lo cual resulta un punto crucial en algunas aplicaciones del álgebra lineal. Esta ventaja motiva encontrar un procedimiento para transformar una base en otra ortogonal. Existe un algoritmo famoso[5] con tal fin llamado *proceso de Gram-Schmidt*. La entrada es una base $\{\vec{v}_1, \vec{v}_2, \dots, \vec{v}_n\}$ que no es ortogonal y la salida otra $\{\vec{u}_1, \vec{u}_2, \dots, \vec{u}_n\}$ que sí lo es. Los pasos del algoritmo son

$$\text{(4.2)} \qquad \vec{u}_1 = \vec{v}_1 \qquad \text{y} \qquad \vec{u}_j = \vec{v}_j - \sum_{k=1}^{j-1} \frac{\langle \vec{v}_j, \vec{u}_k \rangle}{\|\vec{u}_k\|^2}\, \vec{u}_k, \qquad j = 2, 3, \dots, n.$$

En la próxima sección veremos la interpretación de ello en términos de la proyección ortogonal. Comprendiendo esta interpretación no es necesario recordar estas fórmulas (al menos yo no lo hago). Si tras este proceso dividimos $\vec{u}_j$ por $\|\vec{u}_j\|$, lo que se llama *normalización*, los vectores resultantes serán unitarios y formarán una base ortonormal.

Un comentario al margen es que si se aplica el proceso de Gram-Schmidt a vectores $\{\vec{v}_1, \vec{v}_2, \dots, \vec{v}_n\}$ que no son una base porque no son linealmente independientes, entonces el proceso se percata de ello produciendo algún $\vec{u}_j$ nulo.

Para seguir las cuentas veamos primero un ejemplo muy sencillo en que partimos de la base de $\mathbb{R}^3$

$$\mathcal{B} = \{\vec{v}_1 = (2, 0, 0)^t,\ \vec{v}_2 = (1, 0, 3)^t,\ \vec{v}_3 = (3, 5, 7)^t\}.$$

[5] Aunque sea el más famoso y fácil de entender, para cálculos a gran escala con ordenador hay otros más eficientes [4, §9.5].

Según el algoritmo (4.2), tomamos $\vec{u}_1 = (2,0,0)^t$ como paso inicial y el siguiente, correspondiente a $j=2$, produce

$$\vec{u}_2 = \vec{v}_2 - \frac{\langle \vec{v}_2, \vec{u}_1\rangle}{\|\vec{u}_1\|^2}\,\vec{u}_1 = \begin{pmatrix}1\\0\\3\end{pmatrix} - \frac{2}{4}\begin{pmatrix}2\\0\\0\end{pmatrix} = \begin{pmatrix}0\\0\\3\end{pmatrix}.$$

Finalmente para $j=3$ se obtiene

$$\vec{u}_3 = \vec{v}_3 - \frac{\langle \vec{v}_3, \vec{u}_1\rangle}{\|\vec{u}_1\|^2}\,\vec{u}_1 - \frac{\langle \vec{v}_3, \vec{u}_2\rangle}{\|\vec{u}_2\|^2}\,\vec{u}_2 = \begin{pmatrix}3\\5\\7\end{pmatrix} - \frac{6}{4}\begin{pmatrix}2\\0\\0\end{pmatrix} - \frac{21}{9}\begin{pmatrix}0\\0\\3\end{pmatrix} = \begin{pmatrix}0\\5\\0\end{pmatrix}.$$

Es evidente que $\mathcal{B} = \{\vec{u}_1, \vec{u}_2, \vec{u}_3\}$ es ortogonal. Si queremos una base ortonormal al cambiar $\vec{u}_j$ por $\vec{u}_j/\|\vec{u}_j\|$ obtenemos una reordenación de la base canónica.

Al hilo del ejemplo anterior, ortogonalizar en $\mathbb{R}^n$ o $\mathbb{C}^n$ es un poco superfluo porque disponemos de la base canónica que ya es ortogonal. En general no tenemos nada similar en sus subespacios y el proceso cobra sentido ahí. Por ejemplo, consideremos

$$V = \big\{\vec{x} \in \mathbb{R}^3 \,:\, x_1 + 2x_2 - 5x_2 = 0\big\}.$$

Su dimensión es 2 por el Corolario 2.2.3. Lo usual es que para hallar una base elijamos $x_2 = \lambda$, $x_3 = \mu$ de modo que

$$\vec{x} = \begin{pmatrix}x_1\\x_2\\x_3\end{pmatrix} = \begin{pmatrix}-2\lambda + 5\mu\\ \lambda\\ \mu\end{pmatrix} = \lambda\vec{v}_1 + \mu\vec{v}_2 \quad \text{con} \quad \vec{v}_1 = \begin{pmatrix}-2\\1\\0\end{pmatrix},\ \vec{v}_2 = \begin{pmatrix}5\\0\\1\end{pmatrix}.$$

La base $\{\vec{v}_1, \vec{v}_2\}$ no es ortogonal. Para forzar a que lo sea, después de tomar $\vec{u}_1 = \vec{v}_1$ solo hay que dar un paso más de Gram-Schmidt:

$$\vec{u}_2 = \vec{v}_2 - \frac{\langle \vec{v}_2, \vec{u}_1\rangle}{\|\vec{u}_1\|^2}\,\vec{u}_1 = \begin{pmatrix}5\\0\\1\end{pmatrix} - \frac{-10}{5}\begin{pmatrix}-2\\1\\0\end{pmatrix} = \begin{pmatrix}1\\2\\1\end{pmatrix}.$$

A modo de comprobación, está claro que $\vec{u}_1 \cdot \vec{u}_2 = 0$ y $\vec{u}_2 \in V$. Una base ortonormal sería

$$\Big\{\frac{1}{\sqrt{5}}(-2,1,0)^t,\ \frac{1}{\sqrt{6}}(1,2,1)^t\Big\}.$$

Para practicar con los números complejos, hallemos ahora una base ortogonal del subespacio $\mathcal{L}\big(\{\vec{v}_1, \vec{v}_1\}\big) \subset \mathbb{C}^3$ donde

$$\vec{v}_1 = (2+3i, 1+2i, 2)^t \qquad \text{y} \qquad \vec{v}_2 = (8+7i, -3, -6+2i)^t.$$

Como antes, tras tomar $\vec{u}_1 = \vec{v}_1$, todo se reduce al cálculo

$$\frac{\langle \vec{v}_2, \vec{u}_1\rangle}{\|\vec{u}_1\|^2}\,\vec{u}_1 = \frac{(8+7i)(2-3i) + (-3)(1-2i) + (-6+2i)2}{|2+3i|^2 + |1+2i|^2 + |2|^2} = \frac{22}{22} = 1.$$

Con ello, una base ortonormal es

$$\vec{u}_1 = \vec{v}_1 = (2+3i, 1+2i, 2)^t \qquad \text{y} \qquad \vec{u}_2 = \vec{v}_2 - \vec{u}_1 = (6+4i, -4-2i, -8+2i)^t.$$

Si queremos comprobar el resultado, debemos verificar $\vec{u}_1 \cdot \vec{u}_2$ con

$$(2+3i)(6-4i)+(1+2i)(-4+2i)+2(-8-2i) = (24+10i)+(-8-6i)+(-16-4i) = 0.$$

Finalmente, veamos un ejemplo con un producto escalar en un espacio de polinomios. Aunque parezca muy abstracto, al generalizarlo a $\mathbb{R}_n[x]$ aparece en diversas aplicaciones, por ejemplo en la forma de los orbitales atómicos.

Consideramos la base $\{1, x, x^2\}$ de $\mathbb{R}_2[x]$ (los polinomios reales de grado a lo más dos) con el producto escalar $\langle P, Q\rangle = \int_{-1}^{1} PQ$. Si $\{P_1, P_2, P_3\}$ es la base que se obtiene con el proceso de Gram-Schmidt, de acuerdo con las fórmulas (4.2) debemos tomar $P_1 = 1$ y

$$P_2 = x - \frac{\int_{-1}^{1} 1x\;dx}{\int_{-1}^{1} 1^2\;dx} 1 = x, \quad P_3 = x^2 - \frac{\int_{-1}^{1} 1x^2\;dx}{\int_{-1}^{1} 1^2\;dx} 1 - \frac{\int_{-1}^{1} x^3\;dx}{\int_{-1}^{1} x^2\;dx} x = x^2 - \frac{1}{3}.$$

Si queremos una base ortonormal, tenemos que dividir entre la norma y tras algunos cálculos se obtiene

$$\Big\{ \frac{1}{\sqrt{2}}, \sqrt{\frac{3}{2}}\, x, \sqrt{\frac{5}{8}}(3x^2-1) \Big\}.$$

El cabo suelto teórico que queda es mostrar que el proceso de Gram-Schmidt realmente cumple su cometido.

Proposición 4.2.2. *Al aplicar el algoritmo* (4.2) *a una base* $\{\vec{v}_1, \vec{v}_2, \dots, \vec{v}_n\}$ *de un espacio vectorial* V*, con producto escalar, se obtiene siempre una base ortogonal* $\{\vec{u}_1, \vec{u}_2, \dots, \vec{u}_n\}$ *de* V*. En particular todos los espacios vectoriales de dimensión finita y con producto escalar admiten bases ortogonales.*

Demostración. Los vectores $\vec{u}_1$ y $\vec{u}_2 = \vec{v}_2 - \|\vec{u}_1\|^{-2}\langle \vec{v}_2, \vec{u}_1\rangle \vec{u}_1$ son ortogonales como muestra el cálculo de $\langle \vec{u}_2, \vec{u}_1\rangle$. En general, procedemos por inducción finita, si damos por supuesto que $\vec{u}_1, \dots, \vec{u}_{j-1}$ son ortogonales entre sí entonces $\vec{u}_j$ es ortogonal a cualquiera de ellos, digamos a $\vec{u}_\ell$, ya que

$$\langle \vec{u}_j, \vec{u}_\ell\rangle = \langle \vec{v}_j, \vec{u}_\ell\rangle - \sum_{k=1}^{j-1} \frac{\langle \vec{v}_j, \vec{u}_k\rangle}{\|\vec{u}_k\|^2} \langle \vec{u}_k, \vec{u}_\ell\rangle = \langle \vec{v}_j, \vec{u}_\ell\rangle - \frac{\langle \vec{v}_j, \vec{u}_\ell\rangle}{\|\vec{u}_\ell\|^2} \langle \vec{u}_\ell, \vec{u}_\ell\rangle = 0.$$

Por otro lado, $\vec{u}_1$ está en el subespacio generado por $\vec{v}_1$ ya que ambos vectores son idénticos y en general si $\vec{u}_1, \dots, \vec{u}_{j-1} \in \mathcal{L}\big(\{\vec{v}_1, \dots, \vec{v}_{j-1}\}\big)$ entonces necesariamente $\vec{u}_j \in \mathcal{L}\big(\{\vec{v}_1, \dots, \vec{v}_j\}\big)$, gracias a la fórmula que define el algoritmo. Además no es nulo porque $\vec{v}_j \notin \mathcal{L}\big(\{\vec{v}_1, \dots, \vec{v}_{j-1}\}\big)$ (por ser base). Con ello $\{\vec{u}_1, \vec{u}_2, \dots, \vec{u}_n\}$ genera un subespacio de V y como $n = \dim V$ y son vectores ortogonales no nulos debe ser base de V. $\square$

Exprimiendo el silicio [opcional]. Aparentemente en `matlab/octave` no hay una instrucción directa que lleve a cabo el proceso de Gram-Schmidt, quizá porque no es lo más eficiente para cálculos con ordenador, aunque uno puede encontrar en la red muchas implementaciones con pequeños programas. Si nuestro propósito es ortonormalizar sin importar el método, el comando `orth` aplicado a una matriz ortonormaliza el subespacio generado por sus columnas. Es decir, sirve para hallar bases ortonormales en subespacios presentados como $W = \mathcal{L}(\{\vec{v}_1, \dots, \vec{v}_n\})$ incluso si $\{\vec{v}_1, \dots, \vec{v}_n\}$ no es una base. Un ejemplo relacionado con un subespacio que hemos considerado antes es:

```
% Vectores de partida
v1 = [-2;1;0];
v2 = [5;0;1];
v3 = v1 + v2;
v4 = 2*v1 - 7*v2;

% Las columnas de A generan W
A = [v1,v2,v3,v4];

% Las columnas del resultado son
% base ortonormal de W
orth(A)
```

Por supuesto la tercera y la cuarta columnas de `A` son superfluas pues `v1` y `v2` ya generan el subespacio. La salida que se obtiene es:

0.982980	−0.020408
−0.048520	0.929684
0.177188	0.367792

que difiere de los vectores que habíamos obtenido con el proceso de Gram-Schmidt. El comando `orth` también funciona con números complejos.

En `sagemath` sí existe `gram_schmidt` que aplicado sobre una matriz ortogonaliza sus filas con el proceso de Gram-Schmidt. Entonces con el código

```
# Vectores
v1 = [-2,1,0]
v2 = [5,0,1]
A = matrix(QQ, [ v1, v2 ])

G, M = A.gram_schmidt()

# El resultado son las filas de G
print(G)

# La matriz M cumple A = M*G
```

obtenemos una matriz cuyas filas son $(-2, 1, 0)$ y $(1, 2, 1)$, el mismo resultado que habíamos conseguido nosotros. La matriz `M` almacena los coeficientes que aparecen en el proceso y, como se indica, cumple `A = M*G`. Si queremos una base ortonormal basta indicar `orthonormal=True` en el argumento de `gram_schmidt`. Si hacemos esto en el código anterior obtendremos un error porque no podemos trabajar ya de forma simbólica en $\mathbb{Q}$, representado por `QQ`, debemos reemplazar `QQ` por `RDF` que muestra cálculos con decimales. El comando `gram_schmidt` también funciona con números complejos, en este caso hay que usar `CDF` para los cálculos inexactos.