Uso de polymake: Trabajando con poliedros

Representación de $\mathcal{V}$-poliedros

Internamente, `polymake` representa a los poliedros a través de sus conos de homogeinización. Conocemos que si $P = \operatorname{conv}(V) + \operatorname{cone}(Y) \subset \mathbb{R}^d$, entonces el cono de homogeinización de P está dado por:

$$\operatorname{homog}(P) := \operatorname{cone}\begin{pmatrix} \mathbf{1}^T & \mathbf{0}^T \\ V & Y \end{pmatrix} \subset \mathbb{R}^{d+1}.$$

Los vectores (o puntos) en $\mathbb{R}^{d+1}$ para generar $\operatorname{homog}(P)$ se especifican por medio de la propiedad `POINTS` .

Por ejemplo, definimos a continuación `$p` como la envolvente convexa del conjunto $\{(-1,-1),(-1,1),(1,-1),(1,1),(0,0)\}$:

In [46]:
```
$p=new Polytope(POINTS=>[[1,-1,-1],[1,-1,1],[1,1,-1],[1,1,1],[1,0,0]]);
$p->VISUAL;
```

Consultando la propiedad `VERTICES` podemos determinar cuáles son los puntos no redundantes requeridos para definir el polítopo (es decir, sus vértices):

In [47]:
```
print $p->VERTICES;
```

Out[47]:
```
1 -1 -1
1 -1 1
1 1 -1
1 1 1
```

La propiedad `DIM` nos devuelve la dimensión del polítopo:

In [48]:
```
print $p->DIM;
```

Out[48]:
```
2
```

La propiedad `FACETS` nos devuelve las desigualdades que definen las facetas del polítopo. Al consultarla, `polymake` invoca internamente a algún algoritmo tipo Fourier-Motzkin para transformar la representación $\mathcal{V}$ en la representación $\mathcal{H}$ del polítopo. Con la instrucción `prefer "lrs"` especificamos que se utilice el método lrs para la transformación.

In [49]:
```
prefer "lrs";
print_constraints($p->FACETS);
```

Out[49]:
```
0: -x1 >= -1
1: -x2 >= -1
2: x2 >= -1
3: x1 >= -1
```

Con la propiedad `VERTICES_IN_FACETS` consultamos la incidencia entre vértices y facetas:

In [50]:
```
print ($p->VERTICES_IN_FACETS);
```

Out[50]:
```
{2 3}
{1 3}
{0 2}
{0 1}
```

Agregaremos ahora una parte cónica al poliedro anterior. Según la representación del cono de homogeinización, esto se consigue añadiendo un punto (o vector) con la primera coordenada igual a cero al especificar la propiedad `POINTS` .

En el ejemplo siguiente definimos a `$q` como $\mathrm{conv}(\{(-1,-1),(-1,1),(1,-1),(1,1),(0,0)\}) + \mathrm{cone}(\{(1,1)\})$

In [51]:
```
$q = new Polytope(POINTS=>[[1,-1,-1],[1,-1,1],[1,1,-1],[1,1,1],[1,0,0],[0,1,
1]]);
$q -> VISUAL;
```

Al consultar la propiedad `VERTICES` podemos ver que el nuevo poliedro tiene tres vértices (puntos no redundantes en la combinación convexa) y un rayo (vector no redundante en la combinación cónica). Los vértices tienen la primera coordenada igual a 1, los rayos tienen la primera coordenada igual a 0:

In [52]:
```
print($q->VERTICES);
```

Out[52]:
```
1 -1 -1
1 -1 1
1 1 -1
0 1 1
```

Podemos consultar también las desigualdades que definen las facetas de `$q` , así como la incidencia de facetas en vértices:

In [103]:
```
print_constraints($q->FACETS);
print("---\n");
print ($q->VERTICES_IN_FACETS);
```

Out[103]:
```
0: x1 + 3/4 x2 >= -1
1: 3/2 x2 >= -1
2: x1 - 3/4 x2 >= -1
3: -3/2 x2 >= -1
4: -x1 - 3/4 x2 >= -1
5: -x1 + 3/4 x2 >= -1

---
{3 5}
{4 5}
{2 3}
{1 2}
{0 1}
{0 4}
```

Representación de $\mathcal{H}$-poliedros

Un $\mathcal{H}$-poliedro de la forma $P := \{x \in \mathbb{R}^d \ : \ Ax \leq b\}$ se representa a través se su cono de homogeinización, dado por el sistema de desigualdades:

$$\begin{pmatrix} b & -A \end{pmatrix} \begin{pmatrix} x_0 \\ x \end{pmatrix} \geq \mathbf{0}.$$

Las filas de la matriz $\begin{pmatrix} b & -A \end{pmatrix}$ se especifican por medio de la propiedad `INEQUALITIES` , como se indica a continuación en la definición del poliedro `$r` :

In [105]:
```
$r = new Polytope(INEQUALITIES=>[[1,-1,1],[1,1,-1],[0,1,0],[0,0,1],[0,1,1]]);
print_constraints($r->INEQUALITIES);
```

Out[105]:
```
0: -x1 + x2 >= -1
1: x1 - x2 >= -1
2: x1 >= 0
3: x2 >= 0
4: x1 + x2 >= 0
5: 0 >= -1
```

Consultando la propiedad `FACETS` obtenemos las desigualdades no redundantes del sistema:

In [106]:
```
print_constraints($r->FACETS);
```

Out[106]:
```
0: -x1 + x2 >= -1
1: x1 - x2 >= -1
2: x1 >= 0
3: x2 >= 0
```

In [58]:
```
$r->VISUAL;
```

La propiedad `VERTICES` nos indica los vértices y rayos necesarios para expresar al polítopo en la forma $\mathcal{V}$. Al consultar esta propiedad, `polymake` invoca automáticamente a un algoritmo tipo Fourier-Motzkin para cambiar la representación del poliedro.

Recordar que los vértices tienen la primera coordenada igual a 1, mientras que los rayos tienen la primera coordenada igual a 0.

In [59]:
```
print($r->VERTICES);
```

Out[59]:
```
0 1 1
1 0 0
1 1 0
1 0 1
```

Sumas de Minkowski

Definamos nuevamente un polítopo `$p` como la envolvente convexa de un conjunto de puntos:

In [61]:
```
$p=new Polytope(POINTS=>[[1,-1,-1],[1,-1,1],[1,1,-1],[1,1,1],[1,0,0]]);
$p->VISUAL;
```

Definamos ahora a `$q` como el cono generado por los vectores $\binom{10}{9}$ y $\binom{9}{10}$. Al llamar al constructor `Polytope` es necesario indicar explícitamente entre los puntos y vectores en `POINTS` al vértice del cono, aunque el mismo sea el origen $(0,0)$:

In [107]:
```
$q=new Polytope(POINTS=>[[1,0,0],[0,10,9],[0,9,10]]);
$q->VISUAL;
```

Para calcular la suma de Minkowski entre `$p` y `$q` podemos emplear la función `minkowski_sum` , la misma que retorna como resultado un nuevo poliedro:

In [63]:
```
$m = minkowski_sum($p, $q);
$m -> VISUAL;
```

Podemos ahora consultar las diferentes propiedades del nuevo poliedro creado:

In [64]:
```
print ($m->VERTICES);
```

Out[64]:
```
1 -1 -1
1 -1 1
1 1 -1
0 1 9/10
0 1 10/9
```

In [65]:
```
print_constraints($m->FACETS);
```

Out[65]:
```
0: x2 >= -1
1: x1 >= -1
2: 10/19 x1 - 9/19 x2 >= -1
3: -9/19 x1 + 10/19 x2 >= -1
4: 0 >= -1
```

In [66]:
```
print($m->VERTICES_IN_FACETS);
```

Out[66]:
```
{0 2}
{0 1}
{1 4}
{2 3}
{3 4}
```

Conos de recesión

Dado un poliedro, `polymake` puede emplearse para calcular su cono de recesión. Considerar el último poliedro `$m` que hemos definido:

In [108]:
```
$m->VISUAL;
```

Para recuperar el cono de recesión de `$m` llamamos a la función `recession_cone` :

In [109]:
```
$q2= recession_cone($m);
$q2 -> VISUAL;
```

Al contrario de lo que ocurre con los poliedros en general, este cono se representa sin homogeneización. La propiedad `RAYS` permite consultar los vectores que lo generan:

In [110]:
```
print($q2->RAYS);
```

Out[110]:
```
1 9/10
1 10/9
```

Ejercicio

1. Definamos un polítopo `$h` como un hexágono incrustado en el hiperplano $x_3 = 0$ en $\mathbb{R}^3$:

In [111]:
```
$h=new Polytope(POINTS=>[[1,1,0,0],[1,1/2,2/3,0],[1,-1/2,2/3,0],[1,-1,0,0],
[1,1/2,-2/3,0],[1,-1/2,-2/3,0]]);
$h->VISUAL;
```

Definamos ahora `$c` como el cono tridimensional generado por los vectores $\begin{pmatrix} 0 \\ 1 \\ 10 \end{pmatrix}$, $\begin{pmatrix} 2/3 \\ -1/2 \\ 10 \end{pmatrix}$ y $\begin{pmatrix} -2/3 \\ -1/2 \\ 10 \end{pmatrix}$:

In [112]:
```
$c=new Polytope(POINTS=>[[1,0,0,0],[0,0,1,10],[0,2/3,-1/2,10],[0,-2/3,-1/2,1
0]]);
$c->VISUAL;
```

Definamos `$p` como la suma de Minkowski de `$h` y `$c` :

In [113]:
```
$p=minkowski_sum($h,$s);
$p->VISUAL;
```

Consultemos las desigualdades que definen las facetas de `$p` :

In [98]:
```
print_constraints($p->FACETS);
```

Out[98]:
```
0: -3/2 x2 + 3/20 x3 >= -1
1: x3 >= 0
2: x1 - 3/4 x2 + 3/40 x3 >= -1
3: -x1 - 3/4 x2 + 3/40 x3 >= -1
4: -x1 + 3/4 x2 + 5/48 x3 >= -1
5: x1 + 3/4 x2 + 5/48 x3 >= -1
6: -x1 - 4/9 x2 + 2/45 x3 >= -1
7: x1 - 4/9 x2 + 2/45 x3 >= -1
8: 3/2 x2 + 3/40 x3 >= -1
9: 0 >= -1
```

Consultemos los vértices y rayos de `$p` :

In [114]:
```
print($p->VERTICES);
```

Out[114]:
```
1 1 0 0
1 1/2 2/3 0
1 -1/2 2/3 0
1 -1 0 0
1 1/2 -2/3 0
1 -1/2 -2/3 0
0 0 1 10
0 1 -3/4 15
0 -1 -3/4 15
```

Consultemos las incidencias entre los vértices, rayos y facetas de `$p` :

In [100]:
```
print($p->VERTICES_IN_FACETS);
```

Out[100]:
```
{1 2 6}
{0 1 2 3 4 5}
{2 3 6}
{0 1 6}
{0 4 7}
{3 5 8}
{0 6 7}
{3 6 8}
{4 5 7 8}
{6 7 8}
```

Recuperemos el cono de recesión de `$p` :

In [101]:
```
$c2=recession_cone($p);
$c2->VISUAL;
```

In [ ]:
```

```