Dessin vectoriel : fonctions de base

1. Élément draw

Le module de dessin vectoriel permet de créer des dessins qui sont convertis au format SVG. Pour la version PDF de la page, un fichier LaTeX est créé, contenant des commandes PDF, ce qui permet d'obtenir des figures ayant exactement les mêmes polices de caractères que le corps du texte.

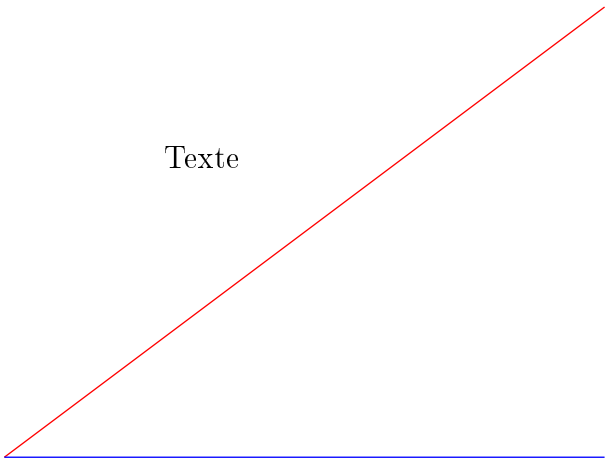
Un dessin vectoriel est défini dans un élément `draw`. La figure SVG apparait dans la page finale à l'emplacement de cet élément. La définition du dessin vectoriel est assez voisine de la norme SVG 1.1 mais comporte les fonctions supplémentaires suivantes :

- ▷ Transformations affines agissant sur les coordonnées des points, et ne modifiant pas l'épaisseur des traits ni la taille des caractères.
- ▷ Position du texte par ancrage (droite, gauche, haut, ...)

Les attributs obligatoires de l'élément `draw` sont `name`, `width` et `height`. Les éléments graphiques définis doivent être compris dans les dimensions ainsi définies, sinon il y a risque de débordements lors de l'affichage en pleine page.

Voici un exemple de figure très simple

```
<draw name="figureA" width="400" height="300">
    <line x1="0" y1="0" x2="400" y2="300" stroke="red"/>
    <line x1="0" y1="0" x2="400" y2="0" stroke="blue"/>
    <text font-size="medium" x="100" y="200" a="l">Texte</text>
</draw>
```



On remarque que le point origine est le point inférieur gauche (y croissant vers le haut), contrairement à SVG et conformément à PDF.

La page PDF a une largeur de 450 points. Par défaut la figure incluse occupe toute la largeur de la page. Pour modifier cette largeur, il suffit de l'indiquer avec l'attribut `pdf-scale`. On remarquera que la taille de la fonte utilisée est indépendante de la largeur de la figure (contrairement à la figure SVG). Avec l'attribut `font-size="medium"`, la fonte est identique à celle du texte, c'est-à-dire une fonte de taille 12 points.

2. Éléments de base

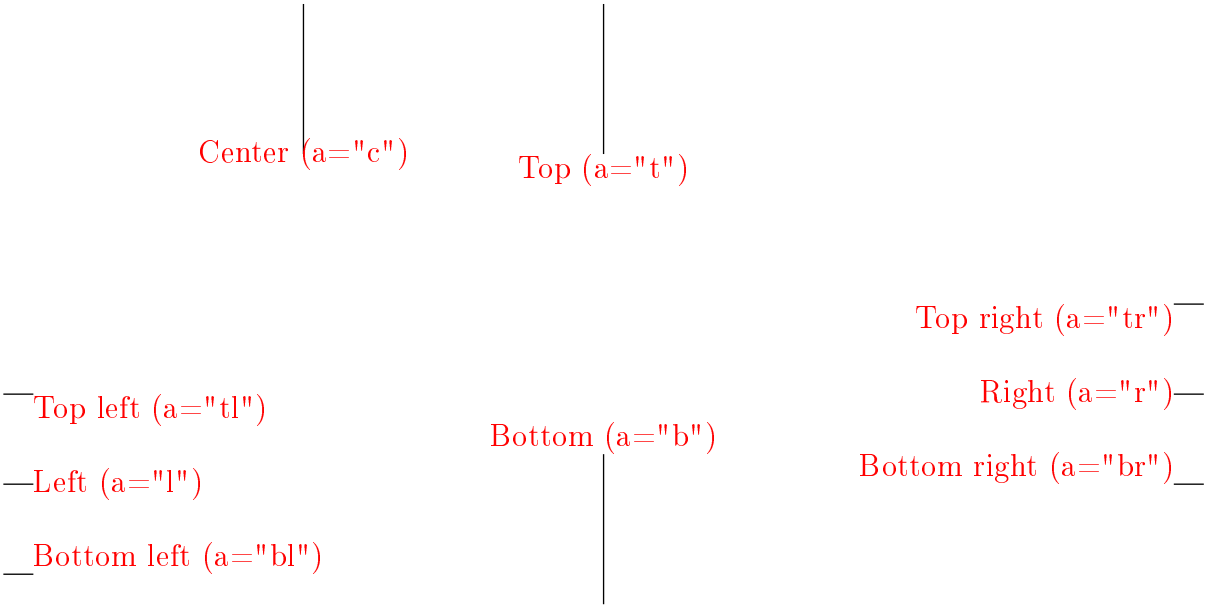
2.a. Élément line

Cet élément est identique à l'élément de même nom de SVG. Ses attributs sont `x1`, `y1`, `x2` et `y2`.

2.b. Élément text

L'élément `text` est similaire à l'élément de même nom de SVG, mais il comporte en plus l'attribut `a` indiquant l'ancrage du texte par rapport au point précisé par les attributs `x` et `y`. La figure suivante montre les ancrages possibles.

```
<draw name="figureB" width="400" height="200" stroke="black" fill="red"
                                        font-size="medium" text-offset="0">
 <line x1="0" y1="10" x2="10" y2="10"/>
 <text x="10" y="10" a="bl">Bottom left</text>
 <line x1="0" y1="40" x2="10" y2="40"/>
 <text x="10" y="40" a="l">Left</text>
 <line x1="0" y1="70" x2="10" y2="70"/>
 <text x="10" y="70" a="tl">Top left</text>
 <line x1="200" y1="200" x2="200" y2="150"/>
 <text x="200" y="150" a="t">Top</text>
 <line x1="200" y1="0" x2="200" y2="50"/>
 <line x1="400" y1="100" x2="390" y2="100"/>
 <text x="390" y="100" a="tr">Top right</text>
 <line x1="400" y1="70" x2="390" y2="70"/>
 <text x="390" y="70" a="r">Right</text>
 <line x1="400" y1="40" x2="390" y2="40"/>
 <text x="390" y="40" a="br">Bottom right</text>
 <line x1="100" y1="200" x2="100" y2="150"/>
 <text x="100" y="150" a="c">Center (a="c")</text>
</draw>
```



Pour laisser un espace entre le point d'ancrage et le texte, on ajoute l'attribut `text-offset`, qui spécifie l'espace en fraction de la taille de la fonte `medium` (valeur par défaut : 0.3)

Voici par exemple une légende placée au dessus et au dessous d'un trait :

```
<draw name="figureC" width="400" height="100" stroke="black" fill="red"
                                    font-size="large" text-offset="0.2">
    <line x1="0" y1="50" x2="400" y2="50"/>
    <text x="200" y="50" a="b">Légende</text>
    <text x="200" y="50" a="t">Dessous</text>
</draw>
```

La taille de la fonte utilisée est indiquée par l'attribut `font-size` dont les valeurs possibles sont (norme CSS) : `medium`, `large`, `x-large`, `xx-large`, `small`, `x-small` et `xx-small`. La taille en points de la fonte `medium` de la figure SVG peut être précisée par l'attribut `medium-font-size` (attribut de `draw`, valeur par défaut 12). Cette taille doit être choisie en fonction de la largeur `width` du dessin, sachant que la fonte garde une proportion constante par rapport à la largeur lorsque la figure est agrandie en pleine page.

Voici par exemple une figure dont la fonte `medium` est 20 fois plus petite que la largeur :

```
<draw name="figureD" width="400" height="100" fill="red"
    font-size="medium" medium-font-size="20" pdf-scale="0.5">
    <polygon points="0,0 400,0 400,100 0,100" fill="none"/>
    <text x="200" y="50" a="c">Texte de taille moyenne</text>
</draw>
```

Dans la page PDF, la fonte `medium` correspond toujours à la fonte du texte, c'est-à-dire 12 points. Si on adopte `pdf-scale="1.0"`, la figure de la page PDF occupe une largeur de 450 points, soit 37.5 fois la taille de la fonte. Pour retrouver les même proportions (largeur du dessin sur taille de fonte) sur la figure SVG, il suffit respecter ce rapport.

Exemple avec toutes les tailles possibles :

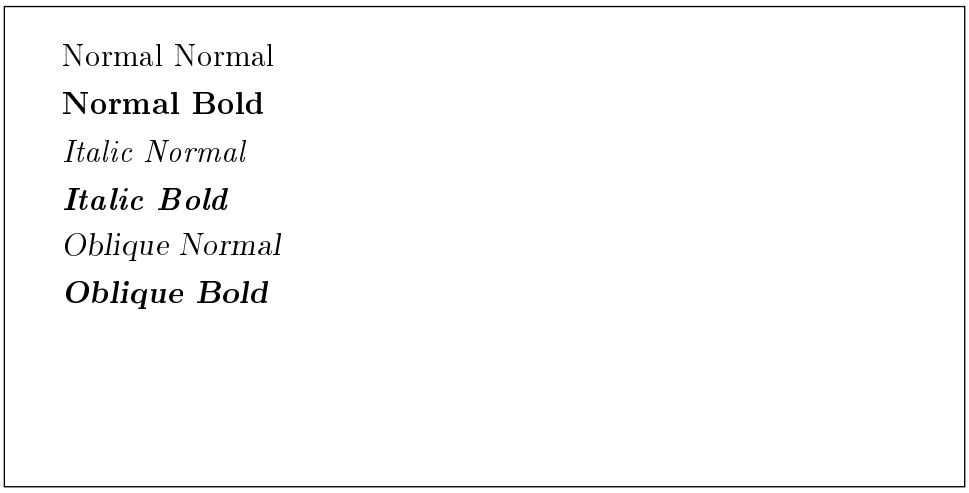
La graisse de la fonte est indiquée par l'attribut `font-weight` pouvant prendre les valeurs `bold` et `normal`.

Le style de la fonte est indiqué par l'attribut `font-style` dont les valeurs possibles sont `normal`, `italic` et `oblique`.

La police de caractère de la figure SVG peut être choisie avec l'attribut `font-family`. En revanche, la police utilisée pour la page PDF est toujours de la même famille (par défaut CMR). En effet les polices utilisées par SVG sont celles du système, alors que les polices utilisées pour la page PDF sont celles de LaTeX.

Afin d'obtenir un rendu uniforme sur tous les systèmes (Windows, MacOS et Linux), il est conseillé d'utiliser les familles de police génériques.

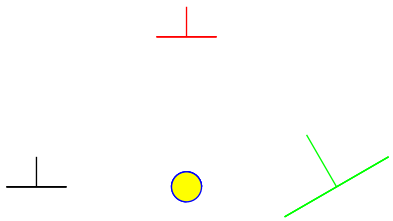
Voici une figure montrant les différents styles et graisses :



On remarquera que les styles `italic` et `oblique` sont souvent identiques sur la figure SVG, alors qu'il sont très différents sur la page PDF (police CMR).

Des équations mathématiques simples peuvent être écrites avec la syntax TeX ; elles doivent être encadrées par des symboles dollars. Actuellement, seuls les symboles standard de LaTeX sont convertis en SVG. Les constructions possibles sont les indices et les exposants. Exemple :

```
<draw name="figureK" width="200" height="100" font-size="medium"
                                    fill="black" pdf-scale="0.5">
    <text x="100" y="70" a="l">$T_{\alpha\beta}$</text>
    <text x="100" y="50" a="l">$\lambda_1$</text>
    <text x="100" y="20" a="l">$C+O_2\rightarrow CO_2$</text>
</draw>
```



La couleur du texte est définie par l'attribut `fill`. Par défaut, l'attribut `stroke` n'a aucun effet sur le texte (conformément à PDF 1.4 et contrairement à SVG 1.1). Dans certains cas, il est intéressant d'attribuer au texte la couleur des dessins (des traits) auquels il est associé, en utilisant l'attribut `stroke`. Pour cela, il suffit de définir l'attribut `fill='stroke'` pour l'élément `text` :

```
<draw name="figureK2" width="200" height="100" font-size="medium" fill="none" stroke=
    <line x1="0" y1="50" x2="200" y2="50"/>
    <text x="100" y="50" a="b" fill="stroke">texte</text>
</draw>
```

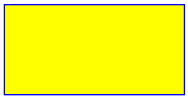
texte

2.c. Éléments polyline et polygon

Ces éléments sont identiques à ceux de SVG 1.1.

Voici par exemple le tracé d'un rectangle :

```
<draw name="figureF" width="200" height="200" pdf-scale="0.3">
    <polygon points="50,100 150,100 150,150 50,150" stroke="blue" fill="yellow"/>
</draw>
```



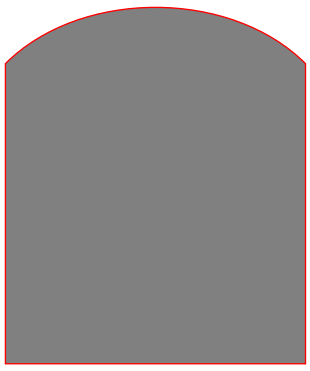
2.d. Élément path

L'élément `path` permet de tracer une courbe. Il contient les éléments suivants :

- ▷ `moveto`
- ▷ `lineto`
- ▷ `curveto` : courbe de Bezier cubique avec deux points de contrôle.
- ▷ `closepath`

Voici un exemple :

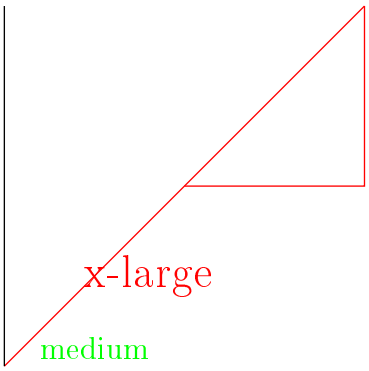
```
<draw name="figureG" width="400" height="400" pdfscale="0.5">
    <path stroke="red" fill="gray">
        <moveto>100,100</moveto>
        <lineto>300,100</lineto>
        <lineto>300,300</lineto>
        <curveto>250,350 150,350 100,300</curveto>
        <closepath/>
    </path>
</draw>
```



2.e. Élément g

L'élément g est identique à l'élément de même nom de SVG 1.1. Il permet de regrouper des éléments ayant des attributs en commun. Voici un exemple :

```
<draw name="figureH" width="200" height="200" pdf-scale="0.3" stroke="black" font-siz
    <g stroke="red" fill="none" font-size="x-large">
        <polyline points="100,100 200,100 200,200"/>
        <line x1="0" y1="0" x2="200" y2="200"/>
        <text x="80" y="50" a="c" fill="red">x-large</text>
    </g>
    <line x1="0" y1="0" x2="0" y2="200"/>
    <text x="50" y="10" a="c" fill="green">medium</text>
</draw>
```



3. Transformations affines

Ces transformations affines agissent sur les coordonnées utilisées dans les éléments `line`, `text`, `polyline`, `polygon` et `path`. Elles n'ont aucun effet sur l'épaisseur des traits et la taille des caractères.

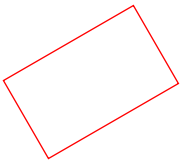
Les transformations sont obtenues par les éléments suivants :

- `translate`, dont les attributs sont `tx` et `ty`.
- `scale`, dont les attributs sont `sx` et `sy`.
- `rotate`, dont l'attribut `a` indique l'angle en degrés.
- `plotrange`, dont les attributs sont `xmin`, `xmax`, `ymin`, `ymax`, `width` et `height`.
- `matrix`, dont l'attribut `m="a,b,c,d,e,f"` définit une transformation affine (composition avec la transformation affine courant).

Ces éléments peuvent contenir des attributs généraux, par exemple `stroke`. Voici un exemple :

```
<draw name="figureI" width="200" height="200" pdf-scale="0.5">
    <translate tx="100" ty="100">
        <rotate a="30">
            <scale sx="50" sy="30">
                <polygon points="0,0 1,0 1,1 0,1" stroke="red" fill="none"/>
```

```
            </scale>
        </rotate>
    </translate>
</draw>
```



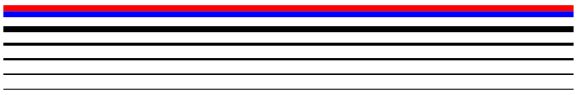
On remarque l'absence de l'élément `ellipse` pourtant présent dans SVG 1.1 (mais pas dans PDF 1.6). Cela est dû au fait qu'une rotation d'une ellipse ne peut être obtenue par une transformation des attributs tels qu'il sont définis dans SVG 1.1. En conséquence, la génération des ellipses et des arcs d'ellipse sera considérée comme une fonction graphique de niveau supérieur. Il en est de même des éléments `circle` et `rect`.

4. Attributs graphiques

4.a. Épaisseur des traits

L'épaisseur des traits est choisie avec l'attribut `stroke-width`. Voici un exemple :

```
<draw name="figureJ" width="400" height="100" pdf-scale="0.5" stroke="black">
    <line x1="10" y1="10" x2="390" y2="10" stroke-width="0.5"/>
    <line x1="10" y1="20" x2="390" y2="20" stroke-width="1"/>
    <line x1="10" y1="30" x2="390" y2="30" stroke-width="1.5"/>
    <line x1="10" y1="40" x2="390" y2="40" stroke-width="2"/>
    <line x1="10" y1="50" x2="390" y2="50" stroke-width="4"/>
    <line x1="10" y1="60" x2="390" y2="60" stroke-width="4" stroke="blue"/>
    <line x1="10" y1="64" x2="390" y2="64" stroke-width="4" stroke="red"/>
</draw>
```



Sur la figure SVG, l'épaisseur du trait est fixée comme la dimension d'un objet. Par exemple, les traits `stroke-width="4"` ci-dessus ont une épaisseur égale à 1/100 de la largeur de la figure. Sur la figure SVG telle qu'elle apparait sur la page XML, la largeur en pixel correspond à la largeur `width` ; le trait le plus fin est donc d'épaisseur 1. Le navigateur Firefox affiche les traits plus fins que 1 pixel avec des niveaux de gris.

Sur la figure PDF, l'épaisseur du trait (en points) est ajusté automatiquement pour retrouver l'épaisseur relative identique à la figure SVG. Lorsqu'on agrandit la figure PDF avec l'attribut `pdf-scale`, l'épaisseur des traits augmente proportionnellement. Ainsi les deux traits rouge et bleu de la figure ci-dessus apparaissent jointifs quelque soit la taille de la figure.

4.b. Traits en pointillé

L'attribut `stroke-dasharray` permet de définir un trait pointillé en donnant la séquence des longueurs à répéter. Lorsqu'il est présent, l'attribut `stroke-dashoffset` indique l'offset de démarrage du trait ; pour la sortie PDF, cet attribut doit être placé avant l'attribut `stroke-dasharray`.

```
<draw name="figureJ1" width="400" height="100" pdf-scale="0.5" stroke="black">
    <line x1="10" y1="10" x2="390" y2="10" stroke-width="1" stroke-dashoffset="5" str
</draw>
```

– — — — — — — — — — — — — —

5. Marqueurs

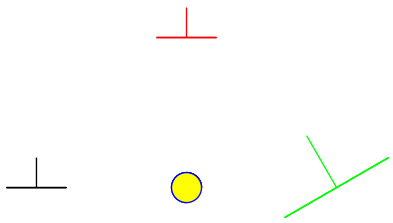
Les marqueurs permettent de dessiner un motif à un emplacement variable, la taille du motif étant indépendante des transformations affines. On peut par exemple tracer un trait faisant toujours 10 unités de long.

Un marqueur se définit dans un élément `marker-def` que l'on place dans un élément `defs`. Un marqueur s'utilise avec l'élément `marker`. Voici un exemple :

```
<draw name="figureK" width="400" height="200" pdf-scale="1" stroke="black" fill="none
    <defs>
        <marker-def id="markerA">
            <marker-moveto>-10,0</marker-moveto>
            <marker-lineto>10,0</marker-lineto>
            <marker-moveto>0,-10</marker-moveto>
            <marker-lineto>0,10</marker-lineto>
        </marker-def>
        <marker-def id="cercle">
            <marker-moveto>10,0</marker-moveto>
            <marker-curveto>10,5.52 5.52,10 0,10</marker-curveto>
            <marker-curveto>-5.52,10 -10,5.52 -10,0</marker-curveto>
```

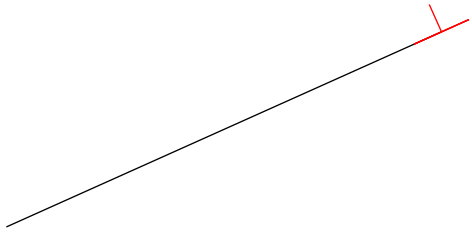
```
            <marker-curveto>-10,-5.52 -5.52,-10 0,-10</marker-curveto>
            <marker-curveto>5.52,-10 10,-5.52 10,0</marker-curveto>
        </marker-def>

    </defs>
    <marker src="#markerA" scale="1.0" x="100" y="100"/>
    <marker src="#markerA" scale="2.0" rotate="30" x="200" y ="100" stroke="green"/>
    <scale sx="1.5" sy="1.5">
        <marker src="#markerA" scale="0.5" x="100" y="100" stroke="red"/>
    </scale>
    <marker src="#cercle" scale="1.0" x="150" y="100" stroke="blue"/>
</draw>
```



Dans cet exemple, le marqueur vert a été tourné de 30 degrés. Il est possible d'effectuer la rotation en indiquant un point définissant une direction avec le point d'ancrage du marqueur :

```
<draw name="figureL" width="400" height="200" pdf-scale="1" stroke="black" fill="none
    <marker-def id="markerA">
        <marker-moveto>-10,0</marker-moveto>
        <marker-lineto>10,0</marker-lineto>
        <marker-moveto>0,0</marker-moveto>
        <marker-lineto>0,10</marker-lineto>
    </marker-def>
    <line x1="10" y1="20" x2="300" y2="150"/>
    <marker src="#markerA" scale="2.0" rotate="10,20" x="300" y="150" stroke="red"/>
</draw>
```



6. Élément clip

Cet élément permet de limiter le tracé de son contenu à l'intérieur d'un rectangle :

```
<draw name="figureM" width="400" height="400" pdf-scale="0.5" stroke="black" fill="no
    <clip x1="100" y1="100" x2="300" y2="300">
        <circle cx="200" cy="200" r="110"/>
    </clip>
</draw>
```