

MATH GAME

MATHÉMATIQUES POUR LES JEUX VIDÉO

ALGORITHMES ET MATHÉMATIQUES



Mathématiques pour les jeux vidéo

Bienvenue dans « Mathgame » !

L'univers du jeu vidéo est une source inépuisable de créativité, où chaque joueur peut explorer des mondes virtuels uniques. Les possibilités sont infinies, mais derrière chaque aspect visuel et mécanique se cachent des fondements mathématiques essentiels.

Dans ce livre, nous vous invitons à plonger dans les mécanismes fondamentaux qui font naître la magie des jeux vidéo. Que vous soyez passionné de jeux, étudiant en sciences ou simplement curieux, ce livre vous permettra de comprendre les principes mathématiques qui donnent vie aux mondes imaginaires que vous explorez.

Des représentations en perspective d'objets, à l'application des textures pour les illuminer, en passant par leur mise en mouvement ou les stratégies pour remporter la victoire, nous explorerons ensemble les concepts mathématiques de base qui sont au cœur de chaque étape de la création d'un jeu.

Le contenu est accessible à tout étudiant ayant obtenu un bac scientifique. Vous pouvez lire les chapitres dans l'ordre que vous voulez : la première partie est la plus mathématique, la deuxième partie est consacrée à la construction des images, la troisième partie fournit des outils pour le mouvement, enfin la quatrième partie introduit la théorie des jeux.

À vos manettes, prêts, partez !

Le livre, l'intégralité des codes ainsi que tous les fichiers sources sont sur la page *GitHub* d'Exo7 :
[« GitHub : Mathgame ».](#)

Sommaire

Résumé des chapitres

Trigonométrie

Savoir mesurer et calculer les angles est fondamental !

Vecteurs

Nous étudions les vecteurs du plan, de l'espace et en n'importe quelle dimension.

Matrices

Les matrices sont des tableaux de nombres très pratiques pour encoder des transformations du plan et de l'espace.

Transformations de l'espace

Nous étudions les transformations affines usuelles de l'espace : translations, homothéties, réflexions... à l'aide des vecteurs et matrices. Nous décrivons les formules de changement de base et introduisons les coordonnées homogènes.

Rotations de l'espace

Nous étudions différentes façons d'obtenir une rotation de l'espace : en la décomposant par des rotations élémentaires ou bien à l'aide des quaternions.

Perspective

Nous expliquons différentes façons de représenter l'espace 3D sur un plan 2D.

Lancer de rayons I

Nous abordons les bases du *ray-tracing* : nous lançons un rayon depuis une source et calculons en quel(s) point(s) ce rayon atteint un objet géométrique (plan, triangle, sphère...).

Lumière

Comment une scène est-elle éclairée ? Quelle est la couleur des objets illuminés ?

Pixels

Comment tracer, pixel par pixel, les figures géométriques de base ?

Texture

Les textures permettent de rendre les objets 3D beaucoup plus réalistes en simulant la couleur et la forme d'une matière. Il s'agit principalement de transformer un carré du plan en une surface de l'espace.

Lancer de rayons II

Nous expliquons en détail la technique du *ray-tracing* et présentons des outils pour accélérer cette méthode.

Triangulation

Nous découpons le plan en objets simples : des triangles.

Maillage

Cette fois nous découpons un objet de l'espace en figures géométriques simples.

Mouvement

Comment se déplacer dans le plan, dans l'espace, dans un labyrinthe, sur un terrain ?

Approximation et interpolation

L'approximation a pour but de modéliser une situation à l'aide d'une fonction simple. Avec une fonction simple, les calculs sont plus rapides. L'interpolation modélise des données partielles par une fonction. On obtient ainsi une fonction qui permet de prédire des valeurs manquantes.

Équations différentielles

Les équations différentielles apparaissent naturellement dans de nombreux domaines au-delà des mathématiques. Elles permettent de modéliser des phénomènes d'évolution en physique, biologie, économie... Nous expliquons ici comment trouver des solutions approchées de ces équations grâce à des méthodes numériques de discréétisation.

Fractales

Les fractales sont des formes géométriques auto-similaires : lorsque l'on zoomé sur une partie, on retrouve une image ressemblant à la figure globale. Les structures fractales permettent de dessiner des paysages et de la végétation. La méthode est facile à implémenter, permet de générer aléatoirement une grande variété de structures, utilise très peu de données, mais par contre nécessite des calculs.

Physique

Pour rendre des animations réalistes, il faut bien comprendre certains principes issus de la physique. Nous en illustrons quelques-uns.

Théorie des jeux

Nous survolons les différents types de jeux, leurs caractéristiques, les différentes stratégies possibles et les équilibres possibles entre adversaires.

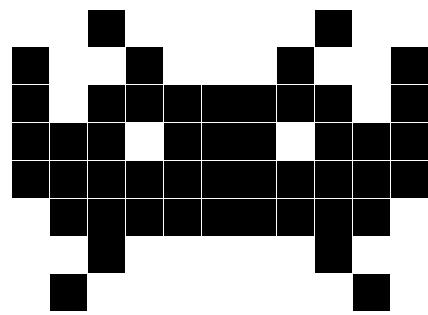
Minimax

L'algorithme minimax permet de choisir le meilleur coup à jouer en anticipant les mouvements de l'adversaire.

Sociologie du joueur

Quel est le comportement d'un joueur dans la « vraie vie » et quels sont les paramètres qui permettent de créer un « bon » jeu ?

PREMIÈRE PARTIE



GÉOMÉTRIE

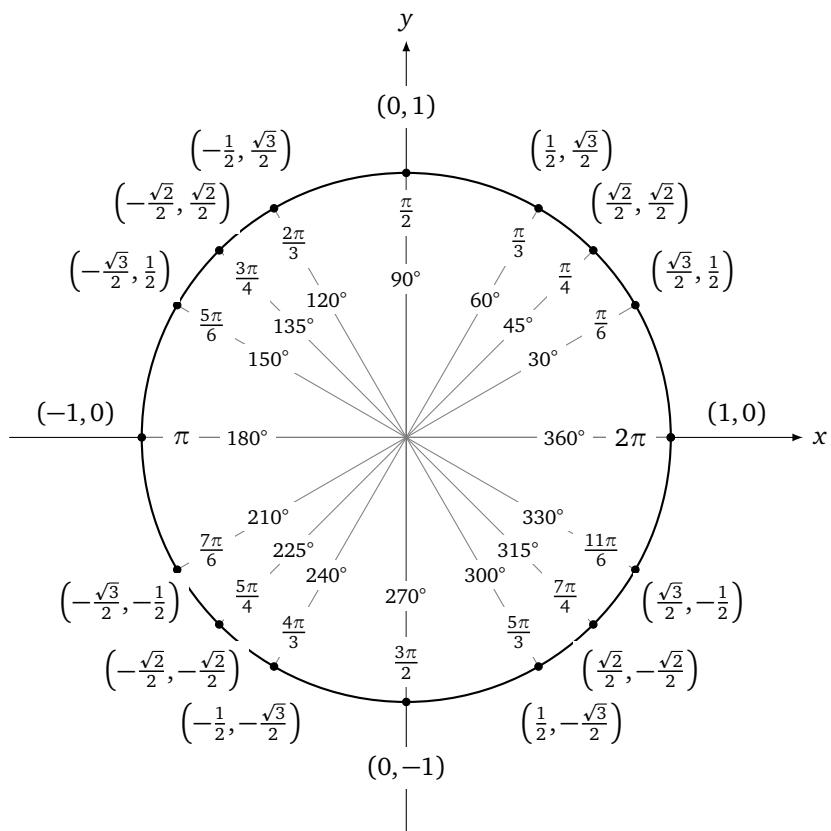
Trigonométrie

Savoir mesurer et calculer les angles est fondamental !

1. Sinus, cosinus, tangente

1.1. Cercle trigonométrique

Le cercle trigonométrique est le cercle centré à l'origine et de rayon 1. Il est orienté dans le sens trigonométrique (le sens inverse des aiguilles d'une montre) à partir du point $(1, 0)$. Sur le cercle ci-dessous, on a placé quelques points avec les angles correspondants en radians (de 0 à 2π) et en degrés (de 0° à 360°) ainsi que leurs coordonnées (x, y) .



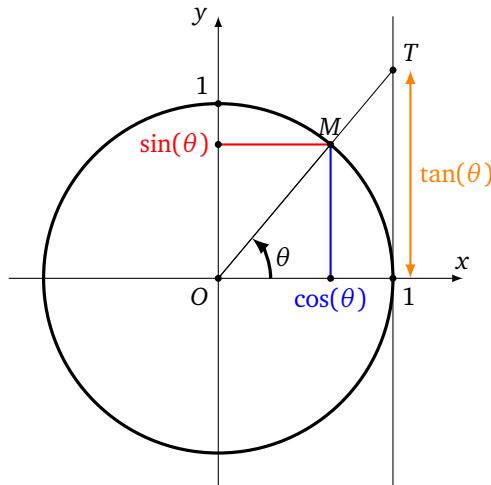
et

$$\theta_{\text{degré}} = 360 \frac{\theta_{\text{radian}}}{2\pi}.$$

Dans ce cours, nous utiliserons principalement le radian comme unité de mesure des angles.

1.2. Sinus, cosinus, tangente

Le point M du cercle trigonométrique correspondant à l'angle θ a pour coordonnées $(\cos(\theta), \sin(\theta))$. Autrement dit, l'abscisse de M est $\cos(\theta)$ et l'ordonnée de M est $\sin(\theta)$.



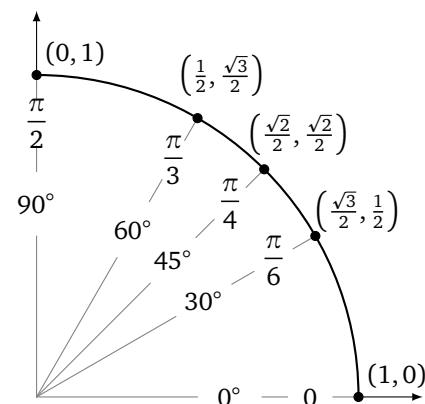
Pour tout θ n'appartenant pas à $\{\dots, -\frac{\pi}{2}, \frac{\pi}{2}, \frac{3\pi}{2}, \frac{5\pi}{2}, \dots\}$ la tangente est définie par

$$\tan(\theta) = \frac{\sin(\theta)}{\cos(\theta)}.$$

La droite (OM) coupe la droite d'équation $(x = 1)$ en T , l'ordonnée du point T est $\tan(\theta)$.

Voici les valeurs des sinus et cosinus pour les angles remarquables.

x	0	$\frac{\pi}{6}$	$\frac{\pi}{4}$	$\frac{\pi}{3}$	$\frac{\pi}{2}$
$\cos(x)$	1	$\frac{\sqrt{3}}{2}$	$\frac{\sqrt{2}}{2}$	$\frac{1}{2}$	0
$\sin(x)$	0	$\frac{1}{2}$	$\frac{\sqrt{2}}{2}$	$\frac{\sqrt{3}}{2}$	1
$\tan(x)$	0	$\frac{1}{\sqrt{3}}$	1	$\sqrt{3}$	



Les formules de base avec sinus et cosinus sont :

$$\cos^2(x) + \sin^2(x) = 1$$

$$\cos(x + 2\pi) = \cos(x)$$

$$\sin(x + 2\pi) = \sin(x)$$

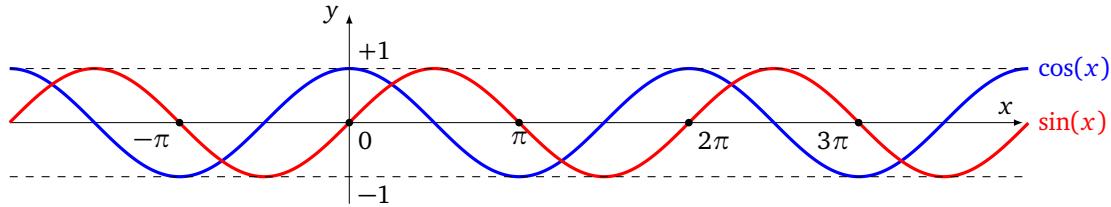
$$\cos(-x) = \cos(x)$$

$$\sin(-x) = -\sin(x)$$

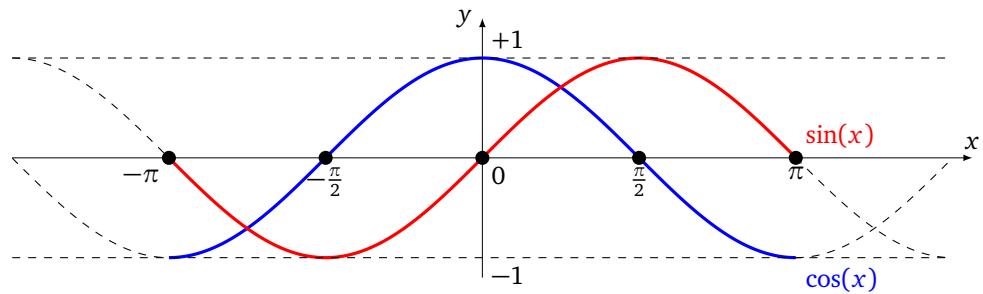
Il en existe beaucoup d'autres !

1.3. Fonctions

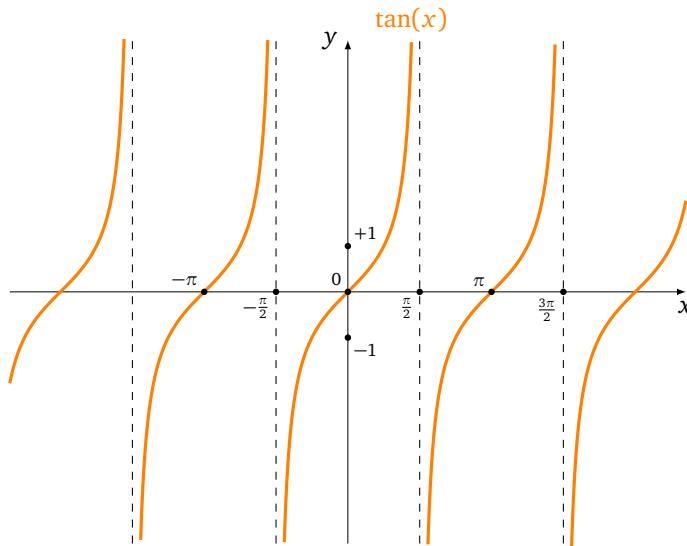
La fonction cosinus $x \mapsto \cos(x)$ est périodique de période 2π et elle est paire (donc symétrique par rapport à l'axe des ordonnées). La fonction sinus $x \mapsto \sin(x)$ est aussi périodique de période de 2π mais elle est impaire (donc symétrique par rapport à l'origine).



Voici un zoom sur l'intervalle $[-\pi, \pi]$.



La fonction $x \mapsto \tan(x)$ est périodique de période π ; c'est une fonction impaire.



Voici les dérivées :

$$\cos'(x) = -\sin(x)$$

$$\sin'(x) = \cos(x)$$

$$\tan'(x) = 1 + \tan^2(x) = \frac{1}{\cos^2(x)}$$

2. Arcsinus, arccosinus, arctangente

Les fonctions trigonométriques inverses permettent de retrouver un angle connaissant la valeur du sinus (ou du cosinus ou de la tangente).

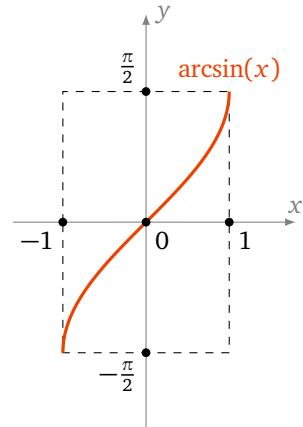
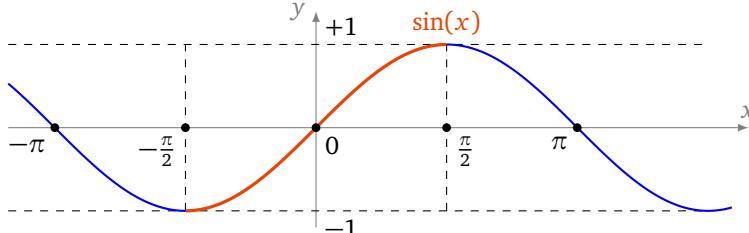
2.1. Arcsinus

Considérons la fonction $\sin : \mathbb{R} \rightarrow [-1, 1]$, $x \mapsto \sin(x)$. Pour obtenir une bijection à partir de cette fonction, il faut considérer la restriction de sinus à l'intervalle $[-\frac{\pi}{2}, \frac{\pi}{2}]$. Sur cet intervalle, la fonction sinus est continue et strictement croissante, donc la restriction

$$\sin| : [-\frac{\pi}{2}, +\frac{\pi}{2}] \rightarrow [-1, 1]$$

est bijective. Sa bijection réciproque s'appelle la fonction **arcsinus** :

$$\arcsin : [-1, 1] \rightarrow [-\frac{\pi}{2}, +\frac{\pi}{2}]$$



On a donc, par définition de la bijection réciproque :

$$\begin{aligned} \sin(\arcsin(x)) &= x & (x \in [-1, 1]) \\ \arcsin(\sin(x)) &= x & (x \in [-\frac{\pi}{2}, +\frac{\pi}{2}]) \end{aligned}$$

Autrement dit :

$$\text{Si } x \in \left[-\frac{\pi}{2}, +\frac{\pi}{2}\right] \quad \sin(x) = y \iff x = \arcsin(y)$$

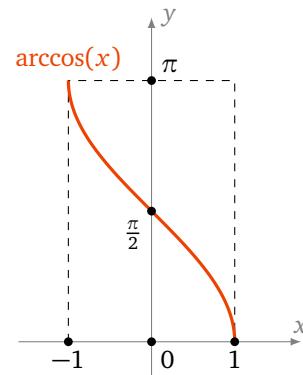
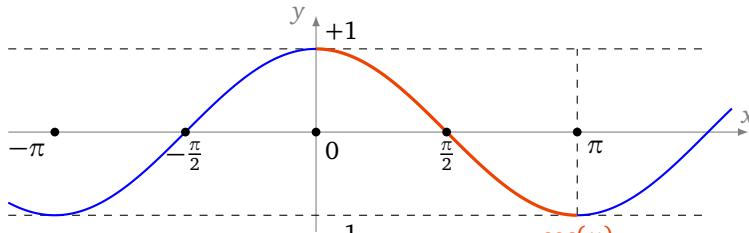
Terminons avec la dérivée de \arcsin :

$$\arcsin'(x) = \frac{1}{\sqrt{1-x^2}} \quad (x \in]-1, 1[)$$

2.2. Arccosinus

La restriction $\cos| : [0, \pi] \rightarrow [-1, 1]$ est une bijection. Sa bijection réciproque est la fonction **arccosinus** :

$$\arccos : [-1, 1] \rightarrow [0, \pi]$$



$$\begin{aligned}\cos(\arccos(x)) &= x & (x \in [-1, 1]) \\ \arccos(\cos(x)) &= x & (x \in [0, \pi])\end{aligned}$$

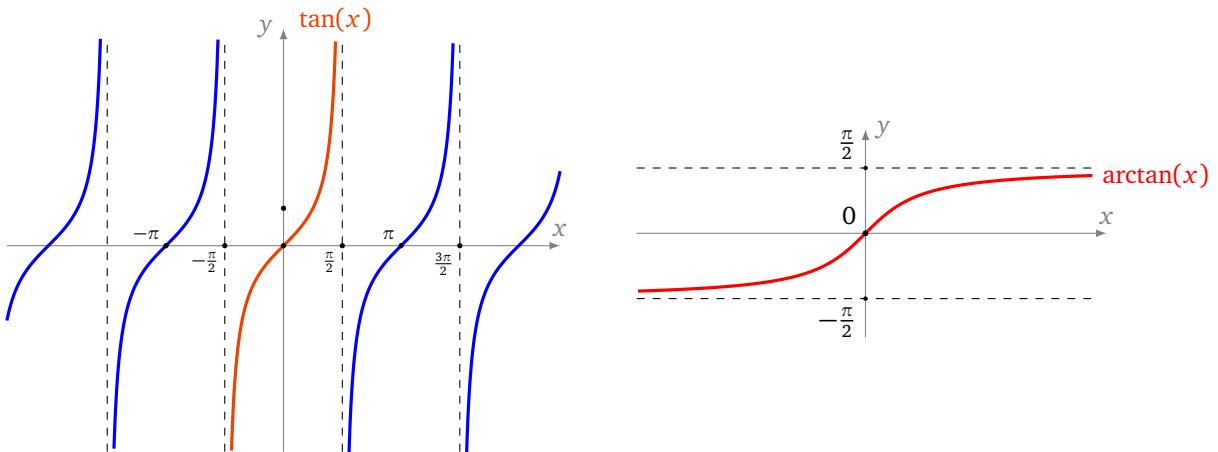
Si $x \in [0, \pi]$ $\cos(x) = y \iff x = \arccos(y)$

$$\arccos'(x) = \frac{-1}{\sqrt{1-x^2}} \quad (x \in]-1, 1[)$$

2.3. Arctangente

La restriction $\tan :]-\frac{\pi}{2}, +\frac{\pi}{2}[\rightarrow \mathbb{R}$ est une bijection. Sa bijection réciproque est la fonction **arctangente** :

$$\arctan : \mathbb{R} \rightarrow]-\frac{\pi}{2}, +\frac{\pi}{2}[$$



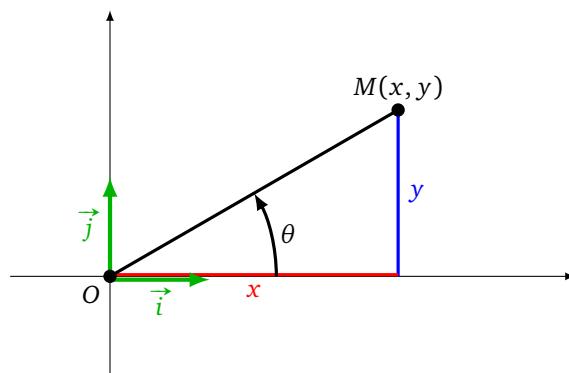
$$\begin{aligned}\tan(\arctan(x)) &= x & (x \in \mathbb{R}) \\ \arctan(\tan(x)) &= x & (x \in]-\frac{\pi}{2}, +\frac{\pi}{2}[)\end{aligned}$$

Si $x \in]-\frac{\pi}{2}, +\frac{\pi}{2}[$ $\tan(x) = y \iff x = \arctan y$

$$\arctan'(x) = \frac{1}{1+x^2} \quad (x \in \mathbb{R})$$

2.4. La fonction arctan2

Détaillons le fonctionnement de la fonction arctan2 qui est essentielle en programmation, mais rarement expliquée dans les cours de mathématiques. L'objectif est tout simplement de retrouver l'angle θ du point M de coordonnées (x, y) (plus précisément l'angle entre \vec{i} qui dirige l'horizontale et \overrightarrow{OM}).



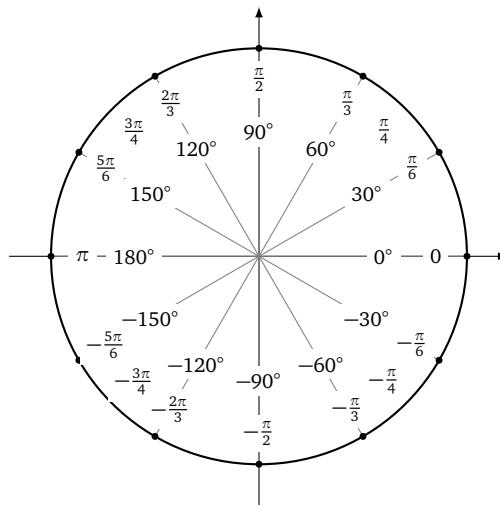
Le cas fondamental est lorsque le point $M(x, y)$ est situé dans le premier quadrant ($x > 0$ et $y \geq 0$) alors :

$$\tan(\theta) = \frac{y}{x},$$

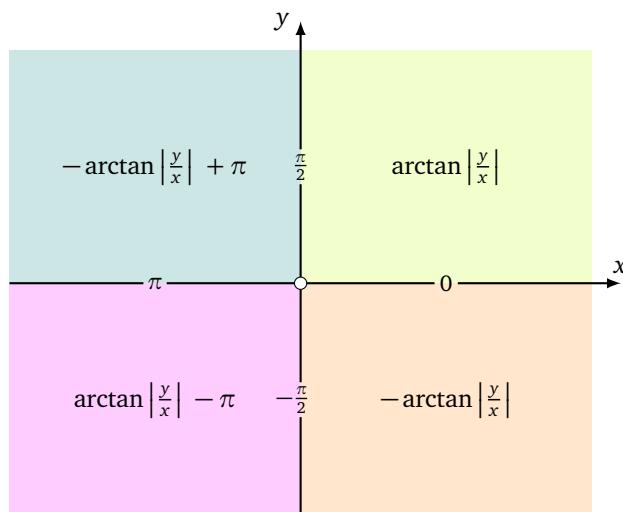
donc

$$\theta = \arctan\left(\frac{y}{x}\right).$$

Il faut adapter le calcul lorsque $M(x, y)$ appartient aux autres quadrants, c'est ce que fait la fonction $\text{arctan}2(y, x)$ (attention à l'ordre des variables!). La fonction $\text{arctan}2(y, x)$ est définie pour tout $(x, y) \neq (0, 0)$ et renvoie l'angle $\theta \in]-\pi, \pi]$ associé au point M avec un angle positif pour les points au-dessus de l'axe des abscisses et un angle négatif en-dessous.



La valeur $\text{arctan}2(y, x)$ se calcule en fonction de $\arctan\left|\frac{y}{x}\right|$, où $\left|\frac{y}{x}\right|$ est la valeur absolue de $\frac{y}{x}$, selon le schéma ci-dessous :



Voici la définition de $\text{arctan}2(y, x)$ à l'intérieur de chacun des quatre quadrants :

$$\text{arctan}2(y, x) = \begin{cases} \arctan\left|\frac{y}{x}\right| & \text{si } x > 0, y > 0 \\ -\arctan\left|\frac{y}{x}\right| + \pi & \text{si } x < 0, y > 0 \\ -\arctan\left|\frac{y}{x}\right| & \text{si } x > 0, y < 0 \\ \arctan\left|\frac{y}{x}\right| - \pi & \text{si } x < 0, y < 0 \end{cases}$$

Sur les axes :

$$\begin{cases} \arctan2(0, x) = 0 & \text{si } x > 0 \\ \arctan2(0, x) = \pi & \text{si } x < 0 \\ \arctan2(y, 0) = \frac{\pi}{2} & \text{si } y > 0 \\ \arctan2(y, 0) = -\frac{\pi}{2} & \text{si } y < 0 \end{cases}$$

On rappelle que la fonction n'est pas définie en $(0, 0)$. Par définition la fonction $\arctan2$ renvoie un angle θ appartenant à $]-\pi, +\pi]$. On peut facilement obtenir un angle de l'intervalle $[0, 2\pi[$: si $\theta < 0$, on change θ en $\theta + 2\pi$.

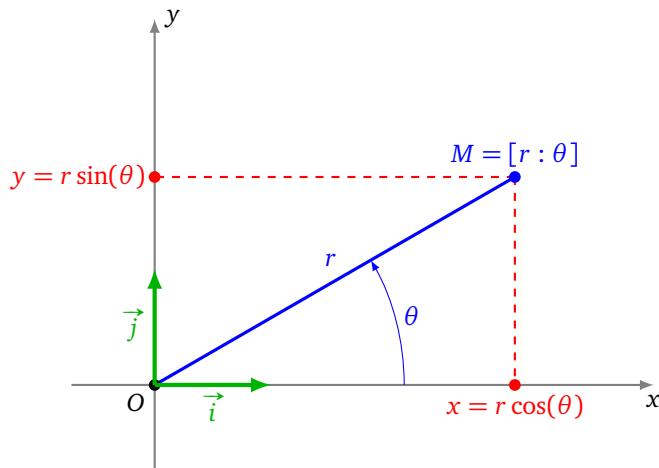
3. Coordonnées polaires

Plutôt que de repérer un point du plan \mathbb{R}^2 par ses coordonnées cartésiennes (x, y) , on peut le faire au moyen de sa distance à l'origine et de l'angle formé avec l'horizontale : ce sont les coordonnées polaires.

3.1. Définition

Soit M un point du plan \mathbb{R}^2 . Soit $O = (0, 0)$ l'origine. Soit (O, \vec{i}, \vec{j}) un repère orthonormé direct.

- On note $r = \|\overrightarrow{OM}\|$, la distance de M à l'origine.
- On note θ l'angle entre \vec{i} et \overrightarrow{OM} .



On note $[r : \theta]$ les **coordonnées polaires** du point M . Dans ce cours, r sera positif. L'angle n'est pas déterminé de manière unique, plusieurs choix sont possibles. Pour avoir l'unicité, on peut limiter θ à l'intervalle $]-\pi, +\pi]$, ou bien $[0, 2\pi[$. On n'attribue généralement pas de coordonnées polaires au point origine (l'angle n'aurait pas de sens).

3.2. Conversion

Coordonnées polaires vers coordonnées cartésiennes.

On retrouve les coordonnées cartésiennes (x, y) à partir des coordonnées polaires $[r : \theta]$ par les formules

$$x = r \cos(\theta) \quad \text{et} \quad y = r \sin(\theta).$$

Coordonnées cartésiennes vers coordonnées polaires.

On retrouve r et θ à partir de (x, y) par les formules suivantes :

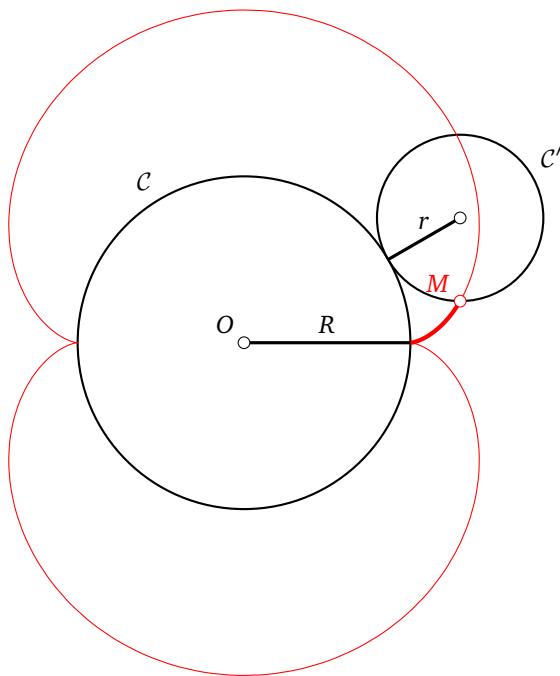
$$r = \sqrt{x^2 + y^2},$$

dans le cas $x > 0$ et $y \geq 0$, $\theta = \arctan\left(\frac{y}{x}\right)$, et dans le cas général
 $\theta = \arctan2(y, x)$.

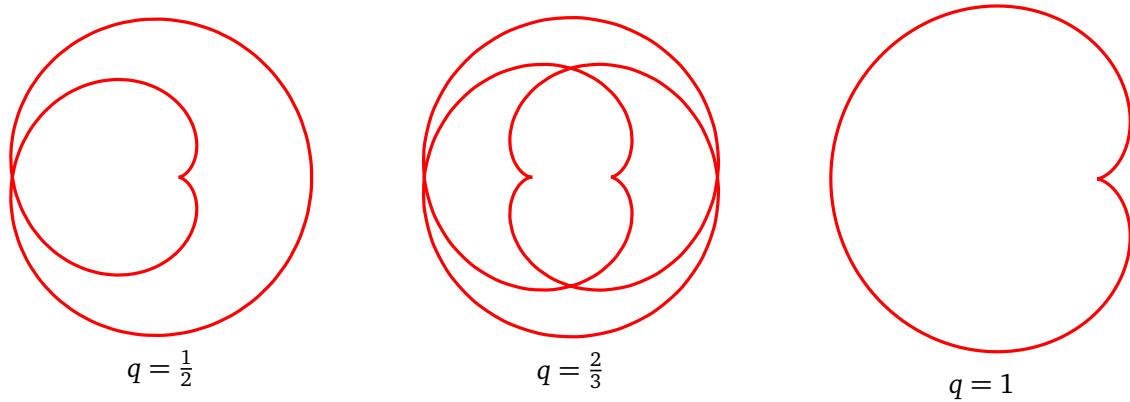
3.3. Exemples

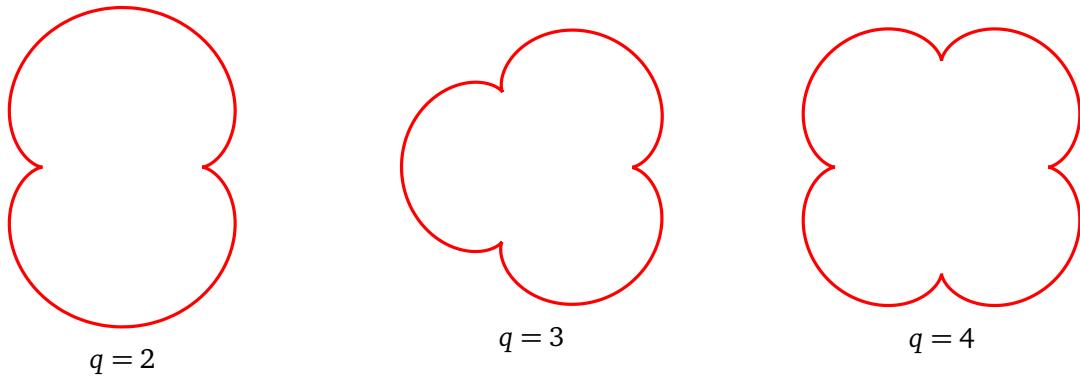
Exemple.

On considère un cercle fixe C centré à l'origine de rayon R , sur lequel roule (sans glisser) un autre cercle C' de rayon r . Sur le cercle C' on choisit un point M . Quelle est la trajectoire de M lorsque le cercle C' roule sur C ?



Cette trajectoire s'appelle une **épicycloïde**, voici plusieurs allures de cette courbe en fonction du rapport $q = \frac{R}{r}$.

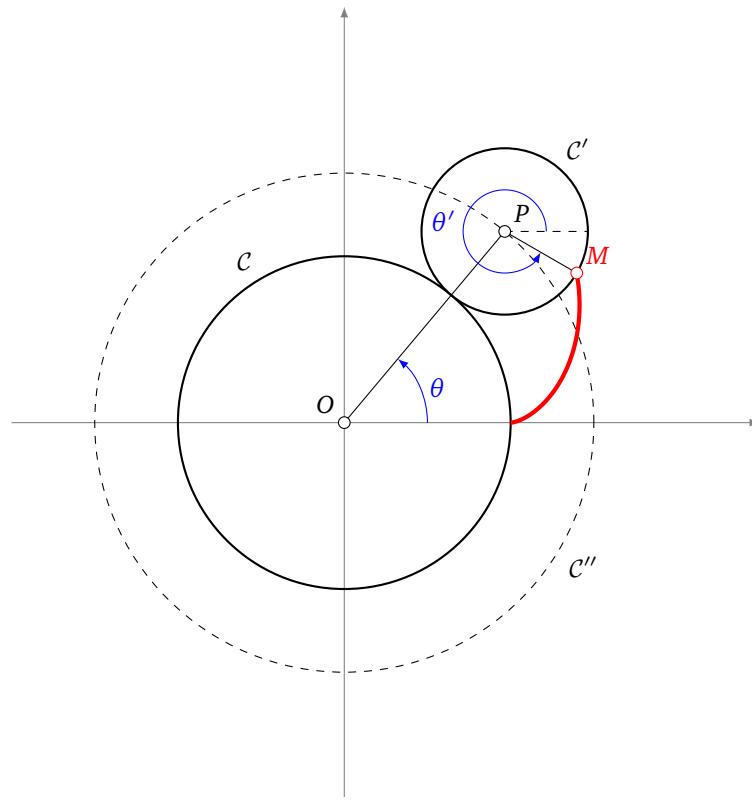




Notons \mathcal{C}'' le cercle (fixe) de centre O et de rayon $R + r$. Le centre (mobile) P du cercle \mathcal{C}' se déplace sur ce cercle \mathcal{C}'' . Les coordonnées du vecteur \overrightarrow{OP} , et donc de P , sont

$$\overrightarrow{OP} = \begin{pmatrix} (R+r) \cos(\theta) \\ (R+r) \sin(\theta) \end{pmatrix},$$

où l'on note θ l'angle entre l'horizontale et \overrightarrow{OP} .



Le vecteur \overrightarrow{PM} a pour coordonnées

$$\overrightarrow{PM} = \begin{pmatrix} r \cos(\theta') \\ r \sin(\theta') \end{pmatrix},$$

où θ' désigne l'angle formé entre \overrightarrow{PM} et l'horizontale. Le roulement sans glissement permet de calculer θ' en fonction de θ ; on donne ici directement la formule :

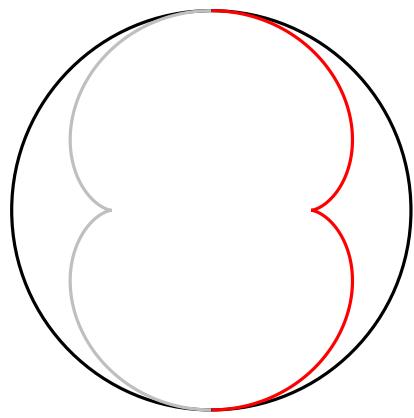
$$\theta' = \pi + \frac{R+r}{r} \theta.$$

Ainsi, les coordonnées de $\overrightarrow{OM} = \overrightarrow{OP} + \overrightarrow{PM}$, et donc du point M , sont :

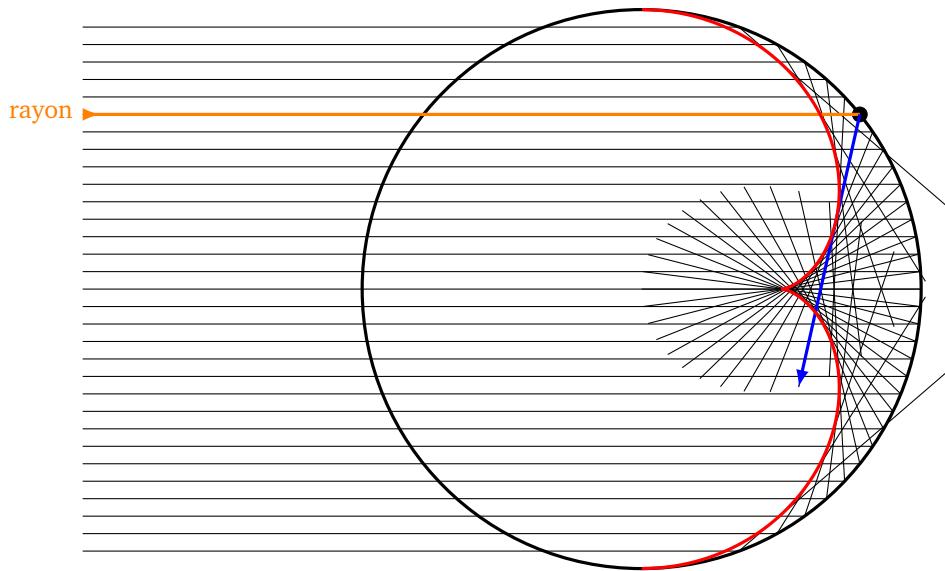
$$\overrightarrow{OM} = \begin{pmatrix} (R+r) \cos(\theta) - r \cos\left(\frac{R+r}{r}\theta\right) \\ (R+r) \sin(\theta) - r \sin\left(\frac{R+r}{r}\theta\right) \end{pmatrix}.$$

Exemple.

Lorsque $R = 2r$, c'est-à-dire le petit cercle roule sur un cercle de rayon deux fois plus grand ($q = 2$ ci-dessus), l'épicycloïde s'appelle la *néphroïde* (elle ressemble à un rein). Mais cette courbe (en fait juste une moitié) a aussi la particularité d'être la *caustique du cercle*.



C'est la courbe visible à la surface du café dans une tasse éclairée par un point éloigné. La caustique correspond aux points où il y a une concentration de lumière. D'un point de vue mathématique et optique, des rayons lumineux parallèles se réfléchissent sur le cercle, la caustique est *l'enveloppe* des rayons réfléchis, c'est-à-dire la courbe tangente aux rayons réfléchis. Ce sont des effets optiques en général très difficiles à calculer, par exemple on ne peut pas les obtenir par *ray-tracing*.



Sur la figure ci-dessus : une source lumineuse est placée à l'infini à gauche, les rayons lumineux arrivent parallèlement et horizontalement. Regardons la trajectoire du rayon incident (orange), il vient intersecer le demi-cercle et se réfléchit ensuite (en bleu) en suivant la loi de réflexion de Descartes. Lorsqu'on fait cela plusieurs fois, les rayons réfléchis dessinent le contour d'une courbe : la caustique.

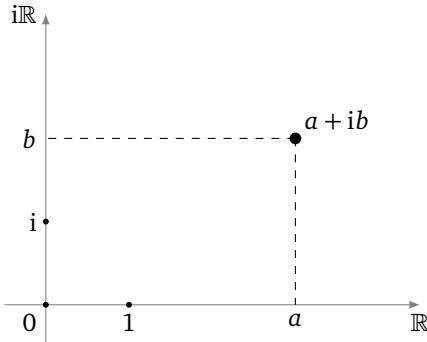
Considérons un cercle \mathcal{C}_0 de rayon $4r$, la caustique de ce cercle a pour équation paramétrique :

$$\overrightarrow{OM} = \begin{pmatrix} 3r \cos(\theta) - r \cos(3\theta) \\ 3r \sin(\theta) - r \sin(3\theta) \end{pmatrix} \quad \theta \in \left[-\frac{\pi}{2}, \frac{\pi}{2}\right].$$

4. Nombres complexes et trigonométrie

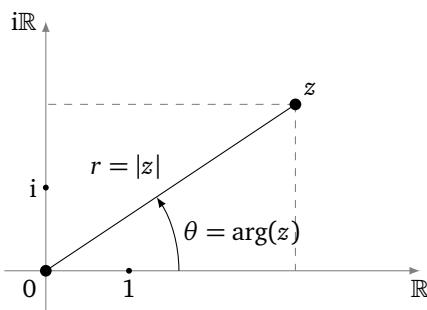
4.1. Écriture trigonométrique

- Un **nombre complexe** est un couple $(a, b) \in \mathbb{R}^2$ que l'on notera $a + ib$.
- Le nombre complexe i vérifie la relation $i^2 = -1$.



- Le **module** de $z = a + ib$ est le réel positif $|z| = \sqrt{a^2 + b^2}$. Il mesure la distance du point (a, b) à l'origine $(0, 0)$.
- Un nombre complexe $z \in \mathbb{C}$, admet l'écriture trigonométrique :

$$z = r \cos(\theta) + ir \sin(\theta) \quad \text{avec } r \in \mathbb{R}_+ \quad \text{et } \theta \in \mathbb{R}$$



- r est en fait le module de z : $r = |z|$,
- θ est un **argument** de z , on le note $\arg(z)$ (en radians).
- Nous définissons la **notation exponentielle** par

$$e^{i\theta} = \cos(\theta) + i \sin(\theta)$$

et donc tout nombre complexe s'écrit :

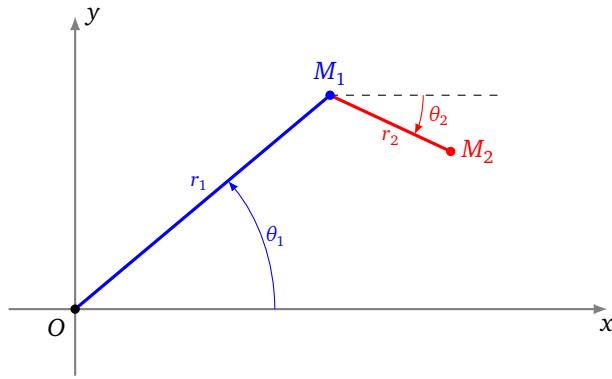
$$z = r e^{i\theta}$$

où $r = |z|$ est son module et $\theta = \arg(z)$ est un de ses arguments.

4.2. Exemple

Exemple.

On considère un système composé de deux bras articulés, le premier peut tourner autour d'un point fixe, le second tourne autour de l'extrémité du premier. Comment calculer la position de l'extrémité du second bras ?



Notons r_1 et r_2 les longueurs (fixes) des deux bras. Les paramètres variables sont les angles θ_1 et θ_2 entre chaque bras et l'horizontale. Le vecteur $\overrightarrow{OM_1}$, et le point M_1 , ont pour affixe :

$$z_1 = r_1 e^{i\theta_1}.$$

Le vecteur $\overrightarrow{M_1 M_2}$ a pour affixe :

$$z_2 - z_1 = r_2 e^{i\theta_2}.$$

Ainsi le vecteur $\overrightarrow{OM_2}$, et le point M_2 , ont pour affixe :

$$z_2 = r_1 e^{i\theta_1} + r_2 e^{i\theta_2}.$$

Si on souhaite retourner aux coordonnées (x, y) de M_2 , ce sont :

$$\begin{pmatrix} r_1 \cos(\theta_1) + r_2 \cos(\theta_2) \\ r_1 \sin(\theta_1) + r_2 \sin(\theta_2) \end{pmatrix}.$$

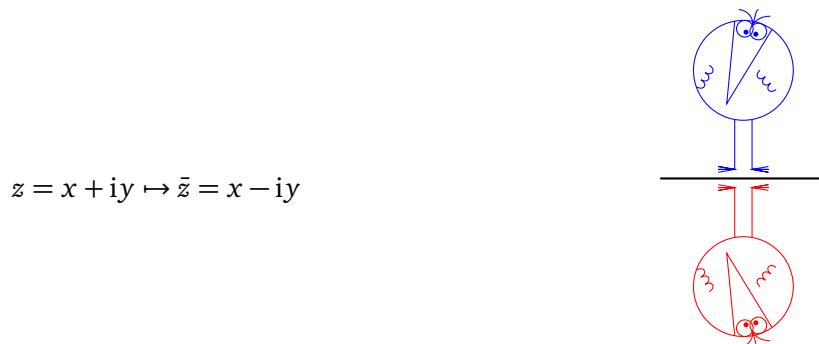
4.3. Transformations géométriques

Voici quelques transformations élémentaires exprimées à l'aide des nombres complexes. On identifie un nombre complexe $z = x + iy$ avec le point $M(x, y)$; ainsi une transformation du plan correspond à une fonction $z \mapsto f(z)$.

- Translation.** La translation de vecteur $\begin{pmatrix} a \\ b \end{pmatrix}$ correspond à l'addition du nombre complexe $z_0 = a + ib$.

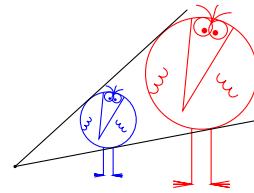


- Réflexion horizontale.** La réflexion par rapport à l'axe des abscisses correspond à la conjugaison complexe.



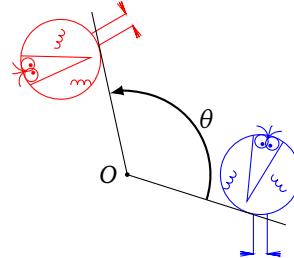
- 3. Homothétie.** L'homothétie centrée à l'origine et de rapport $r \in \mathbb{R}^*$ correspond à la multiplication par le réel r .

$$z \mapsto rz$$



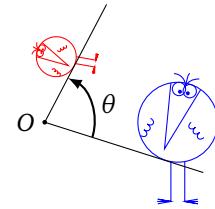
- 4. Rotation.** La rotation de centre l'origine et d'angle θ correspond à la multiplication par $e^{i\theta}$ (un nombre complexe de module 1).

$$z \mapsto e^{i\theta} z$$



- 5. Similitude.** Une similitude directe est la composée d'une homothétie et d'une rotation (ici centrées à l'origine). Elle correspond à la multiplication par un nombre complexe quelconque $w = re^{i\theta}$ avec $r > 0$, où r est le rapport et θ l'angle.

$$z \mapsto wz$$



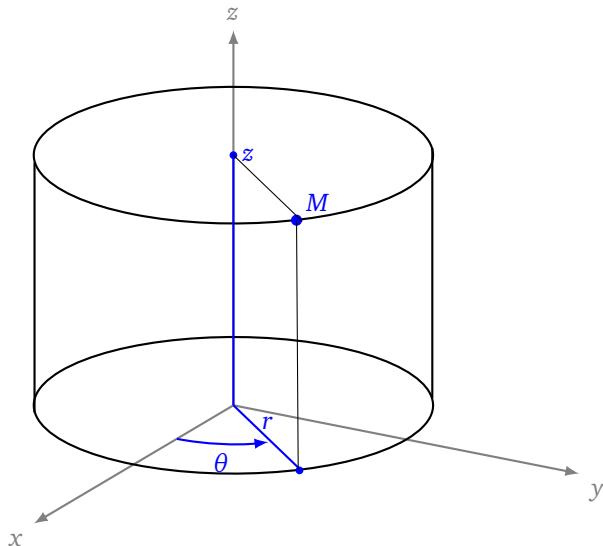
Si on souhaite une similitude centrée en $z_0 = a + ib$ alors la transformation est $z = w(z - z_0) + z_0$.

5. Trigonométrie dans l'espace

5.1. Coordonnées cylindriques

Les coordonnées cylindriques (r, θ, z) sont un autre système de coordonnées que les coordonnées cartésiennes classiques (x, y, z) . On obtient les coordonnées cartésiennes à partir des coordonnées cylindriques par les formules suivantes :

$$\begin{cases} x = r \cos(\theta) \\ y = r \sin(\theta) \\ z = z \end{cases}$$



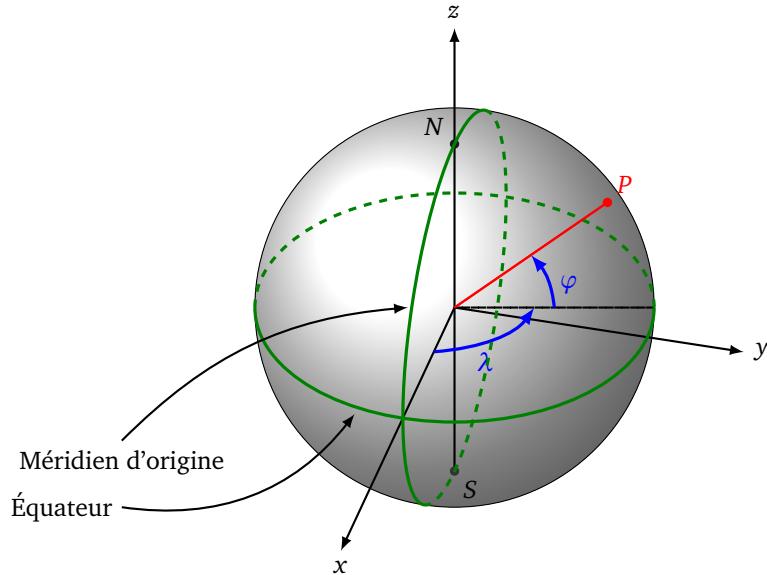
Ces coordonnées sont uniques lorsque $r > 0$ et $\theta \in]-\pi, \pi[$. Elles s'apparentent aux coordonnées polaires avec en plus une hauteur z . Elles sont particulièrement adaptées lorsque la configuration possède un axe de rotation. Les formules inverses sont similaires à celles pour les coordonnées polaires :

$$\begin{cases} r &= \sqrt{x^2 + y^2} \\ \theta &= \arctan2(y, x) \\ z &= z \end{cases}$$

5.2. Coordonnées sphériques

Les coordonnées sphériques (r, φ, λ) sont la donnée d'une altitude r , d'une latitude φ et d'une longitude λ . Elles permettent de se repérer dans l'espace et particulièrement sur une sphère (r , son rayon, est alors fixe). Ce sont les coordonnées naturelles des navigateurs. Le passage des coordonnées sphériques vers les coordonnées cartésiennes s'exprime par :

$$\begin{cases} x &= r \cos(\varphi) \cos(\lambda) \\ y &= r \cos(\varphi) \sin(\lambda) \\ z &= r \sin(\varphi) \end{cases}$$



Ces coordonnées sont uniques si $r > 0$, $\varphi \in]-\frac{\pi}{2}, \frac{\pi}{2}[$, $\lambda \in]-\pi, \pi]$. Attention ! Les notations et les choix d'angles (latitude, colatitude...) diffèrent selon les sources.

Les formules inverses sont :

$$\begin{cases} r &= \sqrt{x^2 + y^2 + z^2} \\ \varphi &= \arcsin\left(\frac{z}{r}\right) \\ \lambda &= \arctan2(y, x) \end{cases}$$

Vecteurs

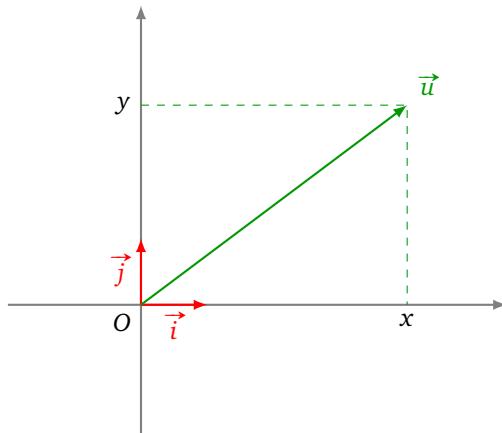
Nous étudions les vecteurs du plan, de l'espace et en n'importe quelle dimension.

1. Vecteurs du plan

1.1. Opérations sur les vecteurs

Dans le plan, on considère (O, \vec{i}, \vec{j}) un repère orthonormé direct. Un **vecteur** \vec{u} est défini par des coordonnées $(x, y) \in \mathbb{R}^2$ par rapport à ce repère. Cela correspond à l'égalité :

$$\vec{u} = x \vec{i} + y \vec{j}.$$



On peut aussi noter les coordonnées verticalement :

$$\vec{u} = \begin{pmatrix} x \\ y \end{pmatrix}.$$

Plusieurs opérations sont définies sur les vecteurs. Soient $\vec{u} = (x, y)$ et $\vec{v} = (x', y')$ deux vecteurs, et soit $\lambda \in \mathbb{R}$ un scalaire.

- L'**addition** de \vec{u} et \vec{v} est définie par

$$\vec{u} + \vec{v} = (x + x', y + y').$$

- La **multiplication par un scalaire** λ est définie par

$$\lambda \vec{u} = (\lambda x, \lambda y).$$

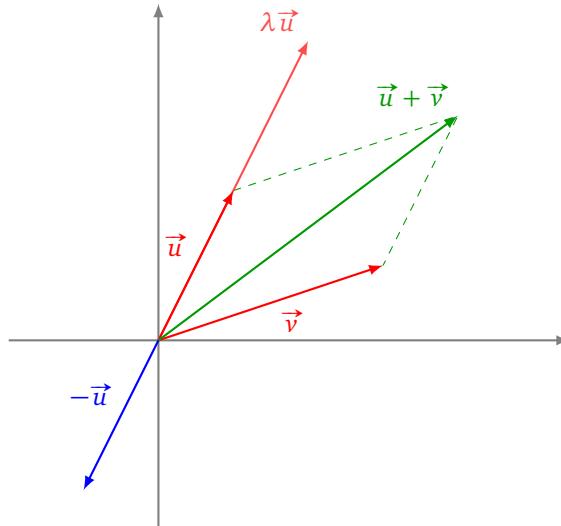
- Le **vecteur nul** est défini par

$$\vec{0} = (0, 0).$$

- Le **vecteur opposé** de \vec{u} est défini par

$$-\vec{u} = (-x, -y).$$

Deux vecteurs \vec{u} et \vec{v} sont **colinéaires** si $\vec{u} = \lambda \vec{v}$ (ou bien $\vec{v} = \lambda \vec{u}$) pour un certain scalaire $\lambda \in \mathbb{R}$.

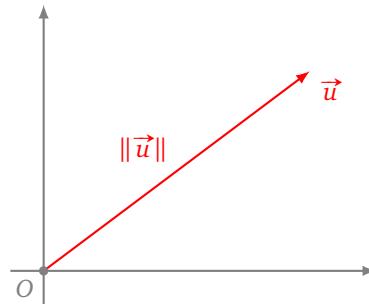


1.2. Norme

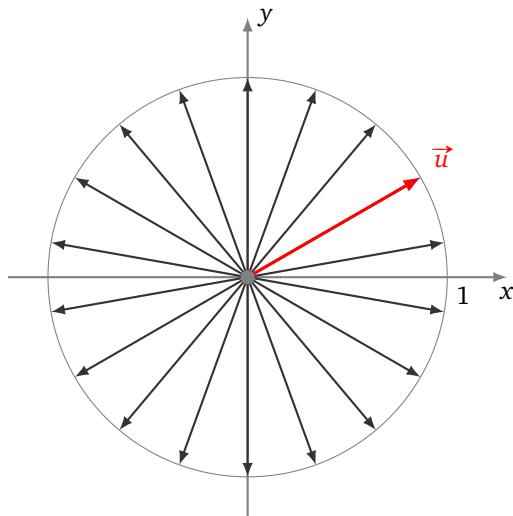
La **norme** d'un vecteur $\vec{u} = (x, y)$ est définie par

$$\|\vec{u}\| = \sqrt{x^2 + y^2}.$$

La norme mesure la distance entre le point $(x, y) \in \mathbb{R}^2$ et l'origine $O = (0, 0)$.



On dira qu'un vecteur est **unitaire** si sa norme vaut 1. Autrement dit, $\vec{u} = (x, y)$ est unitaire si et seulement si $x^2 + y^2 = 1$. Ci-dessous des exemples de vecteurs unitaires.



Soit \vec{u} un vecteur non nul quelconque. La **normalisation** de \vec{u} est le vecteur unitaire $\frac{\vec{u}}{\|\vec{u}\|}$.

1.3. Produit scalaire

Le **produit scalaire** de deux vecteurs \vec{u} et \vec{v} est défini par

$$\vec{u} \cdot \vec{v} = xx' + yy'.$$

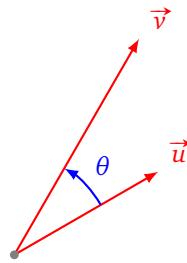
Le produit scalaire mesure à quel point deux vecteurs ont la même direction. On le note aussi $\langle \vec{u} | \vec{v} \rangle$.

Le résultat fondamental est :

Proposition 1.

$$\boxed{\vec{u} \cdot \vec{v} = \|\vec{u}\| \|\vec{v}\| \cos(\theta)}$$

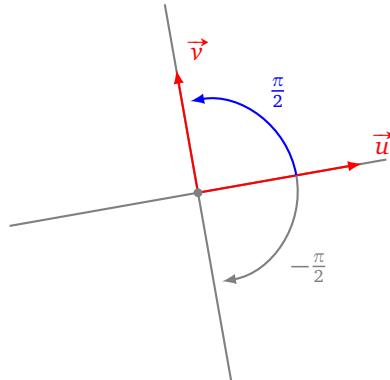
où θ est l'angle entre \vec{u} et \vec{v} .



Cela entraîne :

Proposition 2.

\vec{u} et \vec{v} sont deux vecteurs orthogonaux si et seulement si $\vec{u} \cdot \vec{v} = 0$.

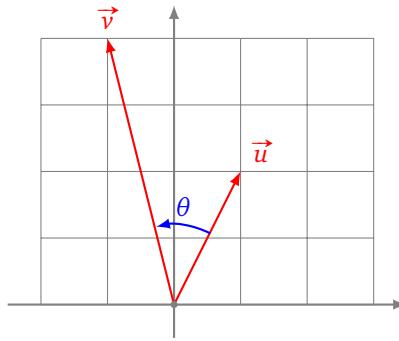


Inversement cette formule permet de calculer l'angle (au signe près) entre deux vecteurs à l'aide du produit scalaire : $\cos(\theta) = \frac{\vec{u} \cdot \vec{v}}{\|\vec{u}\| \|\vec{v}\|}$. La fonction arccos permet de retrouver un angle (sans son signe), connaissant son cosinus. Ainsi l'angle θ , en valeur absolue, vaut :

$$|\theta| = \arccos\left(\frac{\vec{u} \cdot \vec{v}}{\|\vec{u}\| \|\vec{v}\|}\right).$$

Exemple.

Quel est l'angle entre les vecteurs $\vec{u} = (1, 2)$ et $\vec{v} = (-1, 4)$?



On a :

- $\vec{u} \cdot \vec{v} = 1 \times (-1) + 2 \times 4 = 7.$
- $\|\vec{u}\| = \sqrt{1^2 + 2^2} = \sqrt{5}.$
- $\|\vec{v}\| = \sqrt{(-1)^2 + 4^2} = \sqrt{17}.$
- Comme $\vec{u} \cdot \vec{v} = \|\vec{u}\| \|\vec{v}\| \cos(\theta)$ alors $\cos \theta = \frac{7}{\sqrt{5}\sqrt{17}}.$
- Enfin, $\theta = \arccos \frac{7}{\sqrt{5}\sqrt{17}} \simeq 0.708$ radians. Soit $\theta \simeq 40,6^\circ$. (Attention au choix de l'unité d'angle sur votre calculatrice !)

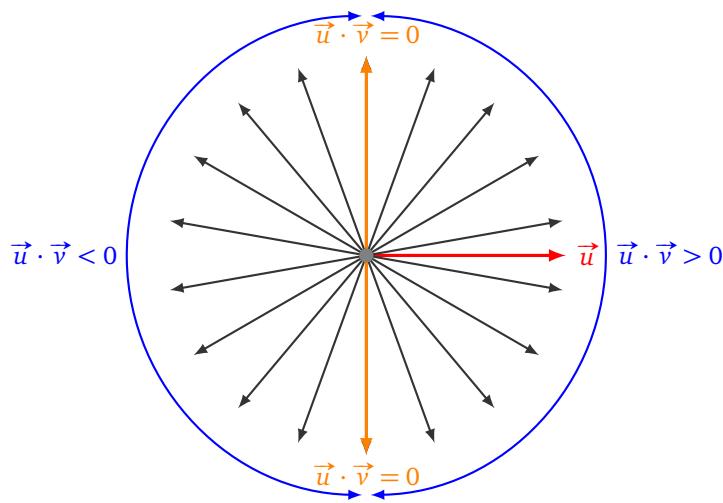
Exemple.

Fixons un vecteur \vec{u} . On considère le vecteur \vec{v} qui est obtenu en tournant \vec{u} d'un angle θ dans le sens trigonométrique. On a alors :

$$\vec{u} \cdot \vec{v} = \|\vec{u}\| \|\vec{v}\| \cos(\theta).$$

- Le produit scalaire est donc nul si et seulement si $\cos \theta = 0$, c'est-à-dire lorsque \vec{u} et \vec{v} sont orthogonaux.
- Le produit scalaire est maximal lorsque $\cos \theta = 1$, c'est-à-dire, \vec{u} et \vec{v} sont colinéaires, dirigés dans le même sens.
- Le produit scalaire est minimal lorsque $\cos \theta = -1$, c'est-à-dire, \vec{u} et \vec{v} sont colinéaires, dirigés dans des sens opposés.

Ci-dessous un vecteur \vec{u} fixé et le signe du produit scalaire $\vec{u} \cdot \vec{v}$ pour différents vecteurs \vec{v} .

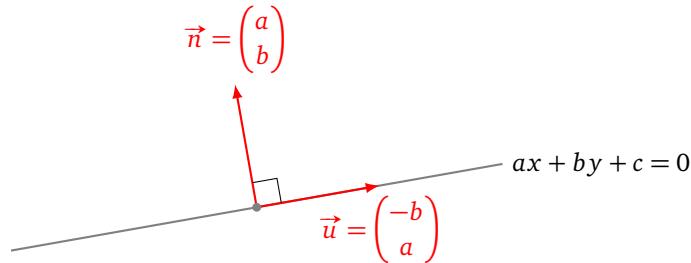


1.4. Applications

Vecteur normal à une droite.

La droite d'équation $ax + by + c = 0$ admet pour vecteur directeur le vecteur $\vec{u} = (-b, a)$. Ainsi un vecteur orthogonal à la droite est le vecteur $\vec{n} = (a, b)$ (on parle aussi de « vecteur normal », sans exiger qu'il soit de norme 1). En effet le produit scalaire de \vec{u} et \vec{n} est nul :

$$\vec{u} \cdot \vec{n} = (-b, a) \cdot (a, b) = -ba + ab = 0.$$

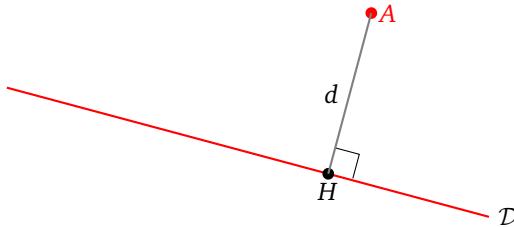


Une application est le résultat suivant.

Proposition 3.

La distance entre un point $A(x_A, y_A)$ quelconque et la droite \mathcal{D} d'équation $ax + by + c = 0$ est donnée par :

$$d(A, \mathcal{D}) = \frac{|ax_A + by_A + c|}{\sqrt{a^2 + b^2}}.$$



Démonstration. $\vec{n} = (a, b)$ est un vecteur orthogonal à la droite \mathcal{D} . Notons H le projeté orthogonal de A sur la droite. La distance d cherchée est la distance AH . Calculons la valeur absolue du produit scalaire de \vec{HA} et \vec{n} de deux manières :

- d'une part \vec{HA} et \vec{n} sont colinéaires, donc

$$|\vec{HA} \cdot \vec{n}| = \|\vec{HA}\| \|\vec{n}\| = d \sqrt{a^2 + b^2}$$

- d'autre part, à l'aide des coordonnées :

$$\vec{HA} \cdot \vec{n} = \begin{pmatrix} x_A - x_H \\ y_A - y_H \end{pmatrix} \cdot \begin{pmatrix} a \\ b \end{pmatrix} = (x_A - x_H)a + (y_A - y_H)b = ax_A + by_A + c$$

On a utilisé que H est un point de la droite \mathcal{D} donc $ax_H + by_H + c = 0$.

On en déduit que :

$$d = \frac{|ax_A + by_A + c|}{\sqrt{a^2 + b^2}}.$$

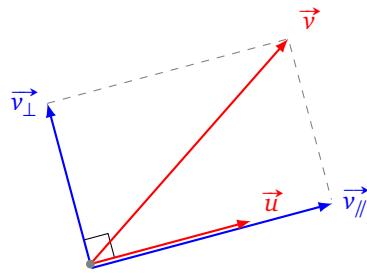
□

Projection orthogonale.

Fixons un vecteur \vec{u} non nul quelconque. Nous pouvons décomposer n'importe quel vecteur \vec{v} en deux parties :

$$\vec{v} = \vec{v}_{\parallel} + \vec{v}_{\perp}$$

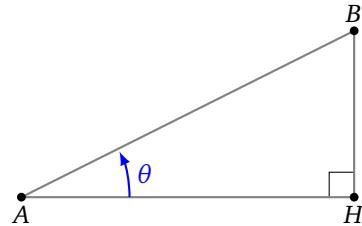
où \vec{v}_{\parallel} est colinéaire à \vec{u} et \vec{v}_{\perp} est orthogonal à \vec{u} .



Le vecteur \vec{v}_{\parallel} est appelé **projeté orthogonal** de \vec{v} sur \vec{u} . Il se calcule par :

$$\vec{v}_{\parallel} = \frac{\vec{u} \cdot \vec{v}}{\|\vec{u}\|^2} \vec{u}.$$

La preuve découle simplement du fait que dans le triangle rectangle suivant on a $AH = AB \cos \theta$.



Le vecteur \vec{v}_{\perp} est alors :

$$\vec{v}_{\perp} = \vec{v} - \vec{v}_{\parallel} = \vec{v} - \frac{\vec{u} \cdot \vec{v}}{\|\vec{u}\|^2} \vec{u}.$$

Exemple.

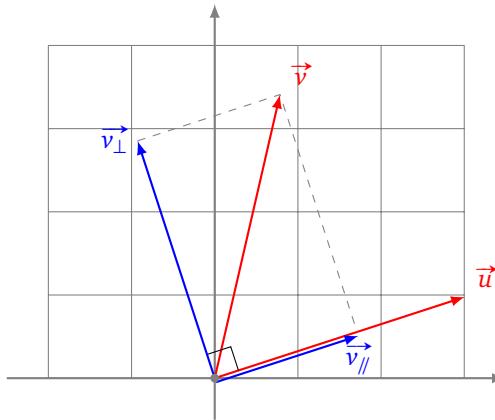
Soit $\vec{u} = (3, 1)$. C'est un vecteur de norme $\|\vec{u}\| = \sqrt{10}$. Soit $\vec{v} = (x, y)$ un vecteur quelconque. On a :

$$\vec{v}_{\parallel} = \frac{\vec{u} \cdot \vec{v}}{\|\vec{u}\|^2} \vec{u} = \frac{1}{10}(3x + y)\vec{u}$$

On pourrait calculer \vec{v}_{\perp} à l'aide de la formule $\vec{v} = \vec{v}_{\parallel} + \vec{v}_{\perp}$.

On peut aussi considérer \vec{u}' , un vecteur orthogonal à \vec{u} , on a : $\vec{u}' = (-1, 3)$. Le projeté orthogonal de \vec{v} sur \vec{u}' est :

$$\vec{v}_{\perp} = \frac{\vec{u}' \cdot \vec{v}}{\|\vec{u}'\|^2} \vec{u}' = \frac{1}{10}(-x + 3y)\vec{u}'$$

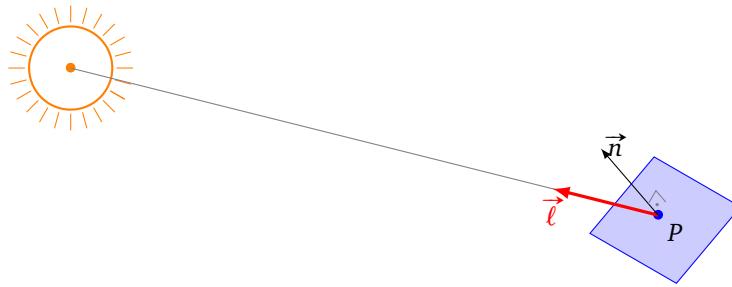


Intensité lumineuse.

L'intensité lumineuse arrivant en P sur un élément de surface est bien sûr proportionnelle à l'intensité i_0 émise, mais elle dépend aussi de l'angle d'incidence.

Notons :

- $\vec{\ell}$: le vecteur unitaire issu de P dirigé vers la source lumineuse,
- \vec{n} : le vecteur unitaire orthogonal à la surface élémentaire.



Alors l'intensité lumineuse i reçue en P est donnée par :

$$i = i_0 \vec{\ell} \cdot \vec{n} = i_0 \cos \theta.$$

où θ est l'angle entre $\vec{\ell}$ et \vec{n} .

Exemple.

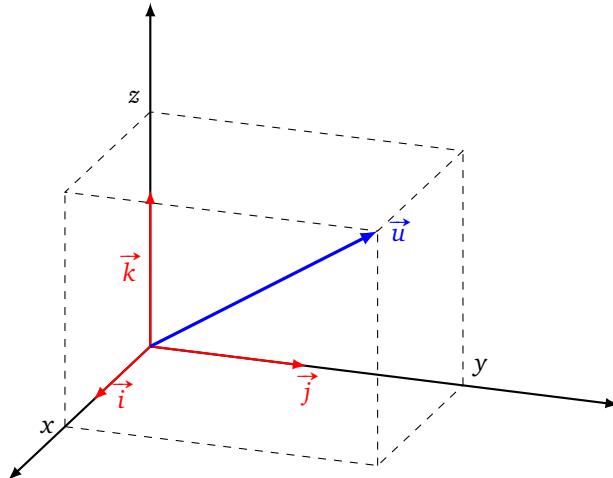
Considérons un angle d'incidence $\theta = \frac{\pi}{4} = 45^\circ$. Alors $i = i_0 \cos \frac{\pi}{4} = i_0 \frac{\sqrt{2}}{2}$. L'intensité reçue est environ 70% de l'intensité émise.

2. Vecteurs dans l'espace

2.1. Opérations sur les vecteurs

Considérons l'espace \mathbb{R}^3 muni du repère orthonormé direct $(O, \vec{i}, \vec{j}, \vec{k})$.

- Un vecteur \vec{u} de l'espace est un triplet (x, y, z) de nombres réels de sorte que $\vec{u} = x \vec{i} + y \vec{j} + z \vec{k}$.



- Opérations. Pour $\vec{u} = (x, y, z)$ et $\vec{v} = (x', y', z')$ et $\lambda \in \mathbb{R}$:

$$\vec{u} + \vec{v} = \begin{pmatrix} x \\ y \\ z \end{pmatrix} + \begin{pmatrix} x' \\ y' \\ z' \end{pmatrix} = \begin{pmatrix} x + x' \\ y + y' \\ z + z' \end{pmatrix} \quad \lambda \vec{u} = \lambda \begin{pmatrix} x \\ y \\ z \end{pmatrix} = \begin{pmatrix} \lambda x \\ \lambda y \\ \lambda z \end{pmatrix}$$

- Le produit scalaire est défini par :

$$\vec{u} \cdot \vec{v} = \begin{pmatrix} x \\ y \\ z \end{pmatrix} \cdot \begin{pmatrix} x' \\ y' \\ z' \end{pmatrix} = xx' + yy' + zz'.$$

- La norme d'un vecteur $\vec{u} = (x, y, z)$ se calcule par :

$$\|\vec{u}\| = \sqrt{\vec{u} \cdot \vec{u}} = \sqrt{x^2 + y^2 + z^2}.$$

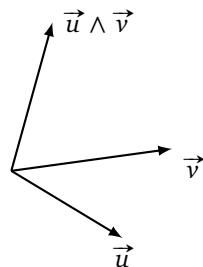
- Deux vecteurs \vec{u} et \vec{v} de l'espace sont orthogonaux si et seulement si $\vec{u} \cdot \vec{v} = 0$.
- L'angle θ (non orienté) entre deux vecteurs \vec{u} et \vec{v} s'obtient par la relation :

$$\cos \theta = \frac{\vec{u} \cdot \vec{v}}{\|\vec{u}\| \|\vec{v}\|}.$$

2.2. Produit vectoriel

Le **produit vectoriel** $\vec{u} \wedge \vec{v}$ est un vecteur orthogonal à \vec{u} et \vec{v} . Il se calcule par la formule :

$$\vec{u} \wedge \vec{v} = \begin{pmatrix} x \\ y \\ z \end{pmatrix} \wedge \begin{pmatrix} x' \\ y' \\ z' \end{pmatrix} = \begin{pmatrix} yz' - zy' \\ zx' - xz' \\ xy' - yx' \end{pmatrix}$$



Vous rencontrerez peut-être aussi la notation anglo-saxonne $\vec{u} \times \vec{v}$ (*cross-product*).

Exemple.

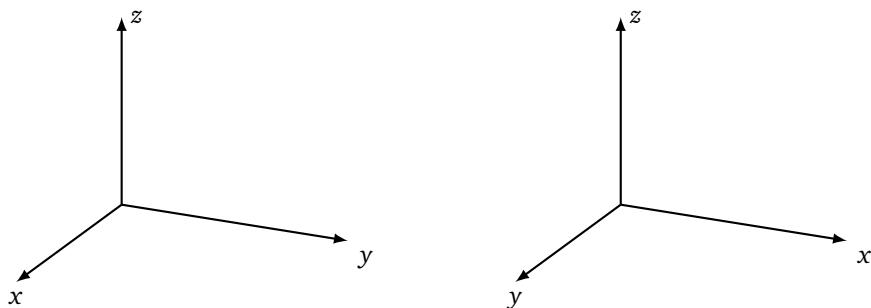
Soient :

$$\vec{u} = \begin{pmatrix} 1 \\ 2 \\ 3 \end{pmatrix} \quad \vec{v} = \begin{pmatrix} 4 \\ 5 \\ 6 \end{pmatrix}$$

Alors :

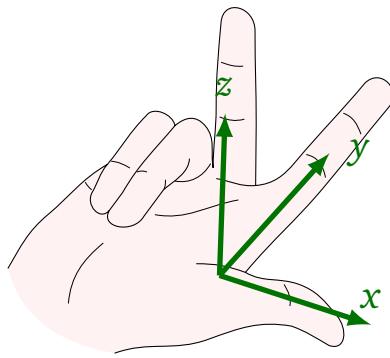
$$\vec{u} \wedge \vec{v} = \begin{pmatrix} 1 \\ 2 \\ 3 \end{pmatrix} \wedge \begin{pmatrix} 4 \\ 5 \\ 6 \end{pmatrix} = \begin{pmatrix} 2 \times 6 - 3 \times 5 \\ 3 \times 4 - 1 \times 6 \\ 1 \times 5 - 2 \times 4 \end{pmatrix} = \begin{pmatrix} -3 \\ 6 \\ -3 \end{pmatrix}$$

Le triplet $(\vec{u}, \vec{v}, \vec{u} \wedge \vec{v})$ est un repère direct de \mathbb{R}^3 (\vec{u} et \vec{v} n'ont pas besoin d'être orthogonaux). On distingue un repère direct de \mathbb{R}^3 d'un repère indirect de \mathbb{R}^3 par la « règle de la main droite ».



Repère direct

Repère indirect



Règle de la main droite

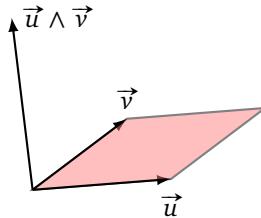
Proposition 4.

- Le produit vectoriel $\vec{u} \wedge \vec{v}$ est un vecteur orthogonal à \vec{u} et à \vec{v} .
- Sa norme vaut :

$$\|\vec{u} \wedge \vec{v}\| = \|\vec{u}\| \|\vec{v}\| |\sin \theta|,$$

où θ est l'angle entre \vec{u} et \vec{v} .

- La norme du produit vectoriel $\vec{u} \wedge \vec{v}$ est égale à l'aire du parallélogramme formé par \vec{u} et \vec{v} .

**Exemple.**

Soient $\vec{u} = (2, -1, -2)$ et $\vec{v} = (3, -1, 1)$.

1. Le produit vectoriel $\vec{w} = \vec{u} \wedge \vec{v}$ est

$$\vec{w} = \begin{pmatrix} 2 \\ -1 \\ -2 \end{pmatrix} \wedge \begin{pmatrix} 3 \\ -1 \\ 1 \end{pmatrix} = \begin{pmatrix} -3 \\ -8 \\ 1 \end{pmatrix}$$

2. On peut vérifier que \vec{w} est orthogonal à \vec{u} et à \vec{v} . En effet : $\vec{u} \cdot \vec{w} = 0$ et $\vec{v} \cdot \vec{w} = 0$.

3. L'aire du parallélogramme (de l'espace) formé par \vec{u} et \vec{v} est :

$$\mathcal{A} = \|\vec{w}\| = \sqrt{(-3)^2 + (-8)^2 + 1^2} = \sqrt{74}$$

Exemple.

Déterminons une équation $ax + by + cz + d$ du plan \mathcal{P} contenant les trois points $A(1, 0, 1)$, $B(1, 3, 0)$ et $C(2, 1, 2)$. Si $\vec{n} = \begin{pmatrix} a \\ b \\ c \end{pmatrix}$ est un vecteur normal à un plan alors une équation de ce plan est $ax + by + cz + d = 0$.

- On calcule les vecteurs $\vec{AB} = (0, 3, -1)$ et $\vec{AC} = (1, 1, 1)$.
- On calcule le produit vectoriel $\vec{n} = \vec{AB} \wedge \vec{AC} = (4, -1, -3)$. C'est un vecteur normal au plan \mathcal{P} . Ainsi $a = 4$, $b = -1$, $c = -3$.
- On détermine d en utilisant les coordonnées d'un point du plan. Par exemple $A \in \mathcal{P}$ donc $ax_A + by_A + cz_A + d = 0$, donc $a \times 1 + b \times 0 + c \times 1 + d = 0$, d'où $d = -1$.
- Conclusion : une équation du plan est $4x - y - 3z - 1 = 0$.

2.3. Produit mixte

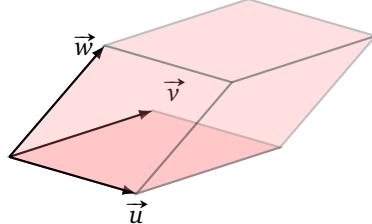
Le **produit mixte** ou **déterminant** des vecteurs \vec{u} , \vec{v} et \vec{w} est le nombre défini par :

$$\det(\vec{u}, \vec{v}, \vec{w}) = \vec{u} \cdot (\vec{v} \wedge \vec{w}).$$

Il est donc formé par un produit vectoriel suivi d'un produit scalaire. Le nom anglais est *triple product*.

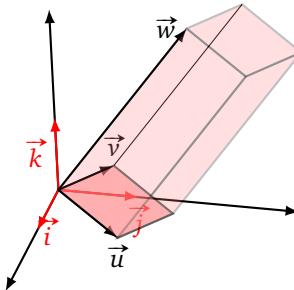
Proposition 5.

Le produit mixte mesure le volume du parallélépipède formé par les trois vecteurs.



Exemple.

Quel est le volume du parallélépipède formé par les vecteurs $\vec{u} = (1, 1, 0)$, $\vec{v} = (1, 1, 1)$ et $\vec{w} = (1, 2, 3)$?



- On calcule le produit vectoriel $\vec{v} \wedge \vec{w} = (1, -2, 1)$.
- On calcule le produit mixte $\det(\vec{u}, \vec{v}, \vec{w}) = \vec{u} \cdot (\vec{v} \wedge \vec{w}) = 1 \times 1 + 1 \times (-2) + 0 \times 1 = -1$.
- Conclusion : le volume du parallélépipède est -1 . Son volume géométrique vaut donc 1 . (Le fait que le volume algébrique soit négatif est dû au fait que les trois vecteurs forment une base indirecte.)

3. Cas général

3.1. Espace vectoriel

On généralise ces notions en considérant des espaces de dimension n pour tout entier positif $n = 1, 2, 3, 4, \dots$. Il n'y a aucune difficulté mathématique excepté le fait qu'il n'est plus possible de visualiser les vecteurs à

partir de la dimension 4. Les éléments de l'espace de dimension n sont les n -uples $\begin{pmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{pmatrix}$ de nombres réels.

L'espace de dimension n est noté \mathbb{R}^n . Comme en dimensions 2 et 3, le n -uple $\begin{pmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{pmatrix}$ dénote aussi bien un point qu'un vecteur de l'espace de dimension n .

Soient $u = \begin{pmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{pmatrix}$ et $v = \begin{pmatrix} y_1 \\ y_2 \\ \vdots \\ y_n \end{pmatrix}$ deux vecteurs de \mathbb{R}^n . L'usage est d'abandonner la flèche pour noter un vecteur.

Définition.

- **Somme de deux vecteurs.** Leur somme est par définition le vecteur $u + v = \begin{pmatrix} x_1 + y_1 \\ \vdots \\ x_n + y_n \end{pmatrix}$.
- **Produit d'un vecteur par un scalaire.** Soit $\lambda \in \mathbb{R}$ (appelé un **scalaire**) : $\lambda u = \begin{pmatrix} \lambda x_1 \\ \vdots \\ \lambda x_n \end{pmatrix}$.
- Le **vecteur nul** de \mathbb{R}^n est le vecteur $0 = \begin{pmatrix} 0 \\ \vdots \\ 0 \end{pmatrix}$.
- L'**opposé** du vecteur $u = \begin{pmatrix} x_1 \\ \vdots \\ x_n \end{pmatrix}$ est le vecteur $-u = \begin{pmatrix} -x_1 \\ \vdots \\ -x_n \end{pmatrix}$.

Théorème 1.

Soient $u = \begin{pmatrix} x_1 \\ \vdots \\ x_n \end{pmatrix}$, $v = \begin{pmatrix} y_1 \\ \vdots \\ y_n \end{pmatrix}$ et $w = \begin{pmatrix} z_1 \\ \vdots \\ z_n \end{pmatrix}$ des vecteurs de \mathbb{R}^n et $\lambda, \mu \in \mathbb{R}$. Alors :

1. $u + v = v + u$
2. $u + (v + w) = (u + v) + w$
3. $u + 0 = 0 + u = u$
4. $u + (-u) = 0$
5. $1u = u$
6. $\lambda(\mu u) = (\lambda\mu)u$
7. $\lambda(u + v) = \lambda u + \lambda v$
8. $(\lambda + \mu)u = \lambda u + \mu u$

Chacune de ces propriétés découle directement de la définition de la somme et de la multiplication par un scalaire. Ces huit propriétés font de \mathbb{R}^n un **espace vectoriel**. Dans le cadre général, ce sont ces huit propriétés qui définissent ce qu'est un espace vectoriel.

3.2. Norme et produit scalaire

- Le **produit scalaire** usuel de $u = (x_1, \dots, x_n)$ et $v = (y_1, \dots, y_n)$, noté $u \cdot v$ (ou bien parfois $\langle u | v \rangle$), est défini par

$$u \cdot v = x_1 y_1 + \cdots + x_n y_n.$$

- La **norme euclidienne** sur \mathbb{R}^n est la norme associée à ce produit scalaire. Pour $u \in \mathbb{R}^n$, la norme euclidienne de u , notée $\|u\|$, est définie par

$$\|u\| = \sqrt{u \cdot u} = \sqrt{x_1^2 + \cdots + x_n^2}.$$

- La **distance** entre le point $A = (a_1, \dots, a_n)$ et le point $M = (x_1, \dots, x_n)$ est

$$\|M - A\| = \sqrt{(x_1 - a_1)^2 + \cdots + (x_n - a_n)^2}.$$

Terminons avec une inégalité qui majore le produit scalaire de deux vecteurs en fonction de leurs normes.

Théorème 2 (Inégalité de Cauchy-Schwarz).

Pour u et v deux vecteurs de \mathbb{R}^n , on a :

$$|u \cdot v| \leq \|u\| \|v\|.$$

3.3. Coordonnées

Définition.

Les vecteurs

$$e_1 = \begin{pmatrix} 1 \\ 0 \\ 0 \\ \vdots \\ 0 \end{pmatrix} \quad e_2 = \begin{pmatrix} 0 \\ 1 \\ 0 \\ \vdots \\ 0 \end{pmatrix} \quad \dots \quad e_n = \begin{pmatrix} 0 \\ 0 \\ \vdots \\ 0 \\ 1 \end{pmatrix}$$

sont appelés les *vecteurs de la base canonique* de \mathbb{R}^n .

Les coordonnées usuelles d'un vecteur $u = \begin{pmatrix} x_1 \\ \vdots \\ x_n \end{pmatrix}$ sont les coordonnées dans la base canonique, c'est-à-dire :

$$u = x_1 e_1 + x_2 e_2 + \dots + x_n e_n.$$

Mais on peut également exprimer des coordonnées du même vecteur u dans une autre base. Une **base** de \mathbb{R}^n est un ensemble $\mathcal{B} = (f_1, f_2, \dots, f_n)$ de n vecteurs, tel que pour tout $u \in \mathbb{R}^n$ il existe des réels y_1, \dots, y_n uniques tels que

$$u = y_1 f_1 + y_2 f_2 + \dots + y_n f_n.$$

$\begin{pmatrix} y_1 \\ \vdots \\ y_n \end{pmatrix}_{\mathcal{B}}$ s'appellent les *coordonnées* du vecteur u dans la base \mathcal{B} .

Exemple.

Soit $\mathcal{B}_0 = (\vec{i}, \vec{j})$ la base canonique de \mathbb{R}^2 (autrement dit (e_1, e_2)). Définissons

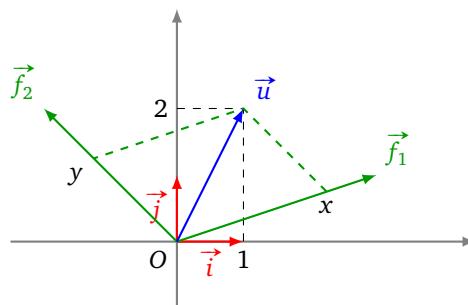
$$\vec{f}_1 = \begin{pmatrix} 3 \\ 1 \end{pmatrix} \quad \vec{f}_2 = \begin{pmatrix} -2 \\ 2 \end{pmatrix}.$$

Alors $\mathcal{B} = (\vec{f}_1, \vec{f}_2)$ est une base de \mathbb{R}^2 .

Soit maintenant le vecteur \vec{u} de coordonnées $\begin{pmatrix} 1 \\ 2 \end{pmatrix}$ dans la base canonique, c'est-à-dire :

$$\vec{u} = 1 \vec{i} + 2 \vec{j}.$$

Quelles sont les coordonnées $\begin{pmatrix} x \\ y \end{pmatrix}_{\mathcal{B}}$ de \vec{u} dans la base \mathcal{B} ?



On veut écrire \vec{u} sous la forme :

$$\vec{u} = x \vec{f}_1 + y \vec{f}_2.$$

On peut donc écrire, avec les coordonnées dans la base canonique :

$$\begin{pmatrix} 1 \\ 2 \end{pmatrix} = x \begin{pmatrix} 3 \\ 1 \end{pmatrix} + y \begin{pmatrix} -2 \\ 2 \end{pmatrix}.$$

On résout le système :

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

On obtient $x = \frac{3}{4}$ et $y = \frac{5}{8}$. On en déduit que les coordonnées de \vec{u} dans la base \mathcal{B} sont :

$$\begin{pmatrix} \frac{3}{4} \\ \frac{5}{8} \end{pmatrix}_{\mathcal{B}}$$

c'est-à-dire :

$$\vec{u} = \frac{3}{4}\vec{f}_1 + \frac{5}{8}\vec{f}_2.$$

Matrices

Les matrices sont des tableaux de nombres très pratiques pour encoder des transformations du plan et de l'espace.

1. Multiplication

1.1. Définition

Une **matrice** A est un tableau rectangulaire de nombres réels. Elle est dite de **taille** $n \times p$ si le tableau possède n lignes et p colonnes. Les éléments de ce tableau sont appelés les **coefficients** de la matrice A . Le coefficient situé à la i -ème ligne et à la j -ème colonne est noté $a_{i,j}$ (ou simplement a_{ij}).

$$A = \begin{pmatrix} a_{1,1} & a_{1,2} & \dots & a_{1,j} & \dots & a_{1,p} \\ a_{2,1} & a_{2,2} & \dots & a_{2,j} & \dots & a_{2,p} \\ \dots & \dots & \dots & \dots & \dots & \dots \\ a_{i,1} & a_{i,2} & \dots & a_{i,j} & \dots & a_{i,p} \\ \dots & \dots & \dots & \dots & \dots & \dots \\ a_{n,1} & a_{n,2} & \dots & a_{n,j} & \dots & a_{n,p} \end{pmatrix} \quad \text{ou} \quad A = (a_{i,j})_{\substack{1 \leq i \leq n \\ 1 \leq j \leq p}} \quad \text{ou} \quad (a_{i,j}).$$

L'ensemble des matrices à n lignes et p colonnes à coefficients réels est noté $M_{n,p}(\mathbb{R})$ ou simplement $M_{n,p}$.

Exemple.

$A = \begin{pmatrix} 4 & -1 \\ 2 & 5 \\ -1 & 0 \end{pmatrix} \in M_{3,2}$ est une matrice 3×2 avec, par exemple, $a_{1,1} = 4$ et $a_{3,2} = 0$.

1.2. Vecteurs

Vecteur ligne/vecteur colonne. Un vecteur de longueur n peut être à la fois vu comme un vecteur colonne ou bien un vecteur ligne. Un vecteur colonne de longueur n est un cas particulier d'une matrice à n lignes et 1 colonne. Un vecteur ligne de longueur n est une matrice à 1 ligne et n colonnes.

$$\begin{pmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{pmatrix} \in M_{n,1} \quad (x_1 \ x_2 \ \dots \ x_n) \in M_{1,n}$$

Matrice comme juxtaposition de vecteurs. Il est pratique de considérer qu'une matrice est la juxtaposition de vecteurs colonnes. Plus précisément une matrice $A \in M_{n,p}$ est composée de p vecteurs colonnes C_1, \dots, C_p chacun de longueur n .

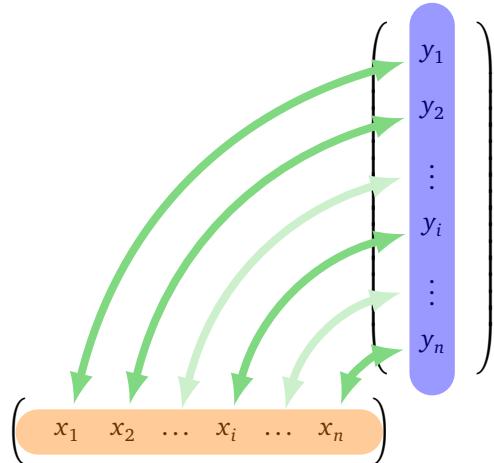
$$\begin{array}{c}
 C_j \\
 \left(\begin{array}{cccccc} a_{11} & a_{12} & \cdots & a_{1j} & \cdots & a_{1p} \\ a_{21} & a_{22} & \cdots & a_{2j} & \cdots & a_{2p} \\ \cdots & \cdots & \cdots & \cdots & \cdots & \cdots \\ a_{i1} & a_{i2} & \cdots & a_{ij} & \cdots & a_{ip} \\ \cdots & \cdots & \cdots & \cdots & \cdots & \cdots \\ a_{n1} & a_{n2} & \cdots & a_{nj} & \cdots & a_{np} \end{array} \right) \\
 L_i \\
 \left(\begin{array}{cccccc} a_{11} & a_{12} & \cdots & a_{1j} & \cdots & a_{1p} \\ a_{21} & a_{22} & \cdots & a_{2j} & \cdots & a_{2p} \\ \cdots & \cdots & \cdots & \cdots & \cdots & \cdots \\ a_{i1} & a_{i2} & \cdots & a_{ij} & \cdots & a_{ip} \\ \cdots & \cdots & \cdots & \cdots & \cdots & \cdots \\ a_{n1} & a_{n2} & \cdots & a_{nj} & \cdots & a_{np} \end{array} \right)
 \end{array}$$

Bien sûr on peut aussi considérer que cette même matrice est la superposition de n vecteurs lignes L_1, \dots, L_n chacune de longueur p .

Produit scalaire. Rappelons que le *produit scalaire* de $u = (x_1, \dots, x_n)$ et $v = (y_1, \dots, y_n)$, noté $u \cdot v$ (ou parfois $\langle u | v \rangle$), est défini par

$$u \cdot v = x_1 y_1 + \cdots + x_n y_n.$$

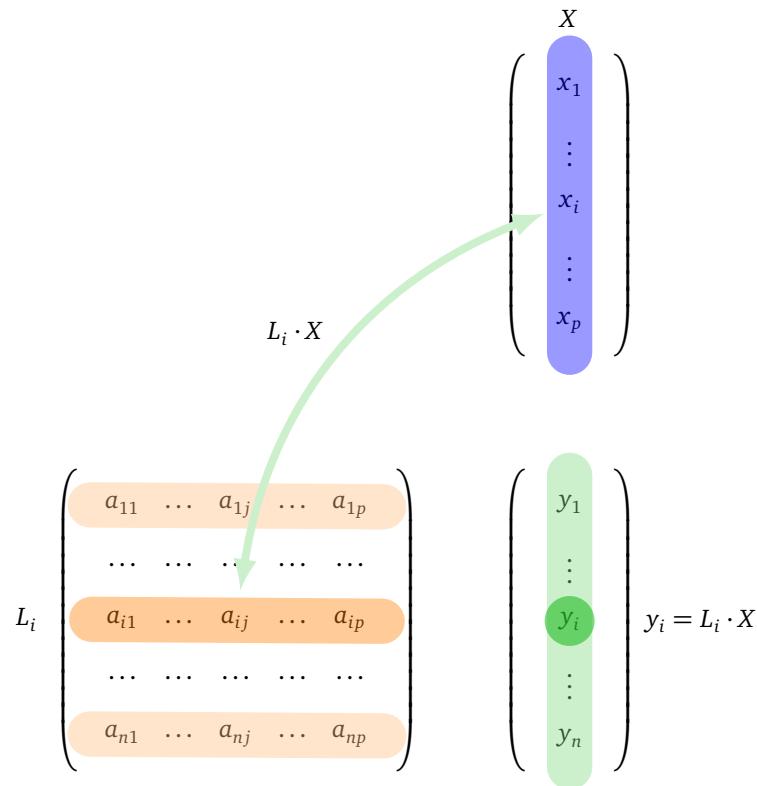
Expliquons le produit scalaire $u \cdot v$ en termes de vecteur ligne/vecteur colonne : on considère u comme un vecteur ligne et v comme un vecteur colonne, puis on multiplie entre eux les deux premiers coefficients, ensuite on multiplie entre eux les deuxièmes coefficients, etc. Le produit scalaire est la somme de tous ces produits.



Multiplication d'une matrice par un vecteur. Soit $A \in M_{n,p}$ une matrice ayant n lignes et p colonnes. Soit X un vecteur de longueur p , considéré comme un vecteur colonne. Le produit AX est un vecteur colonne Y de longueur n défini ainsi :

$$\underbrace{\left(\begin{array}{ccc} a_{11} & \cdots & a_{1p} \\ a_{21} & \cdots & a_{2p} \\ \vdots & & \vdots \\ a_{n1} & \cdots & a_{np} \end{array} \right)}_A \underbrace{\left(\begin{array}{c} x_1 \\ x_2 \\ \vdots \\ x_p \end{array} \right)}_X = \underbrace{\left(\begin{array}{c} a_{11}x_1 + a_{12}x_2 + \cdots + a_{1p}x_p \\ a_{21}x_1 + a_{22}x_2 + \cdots + a_{2p}x_p \\ \vdots \\ a_{n1}x_1 + a_{n2}x_2 + \cdots + a_{np}x_p \end{array} \right)}_Y.$$

Considérons A comme une superposition de vecteur lignes L_1, \dots, L_n . Le premier coefficient de $Y = AX$ est en fait le produit scalaire $L_1 \cdot X$, le deuxième coefficient est $L_2 \cdot X, \dots$

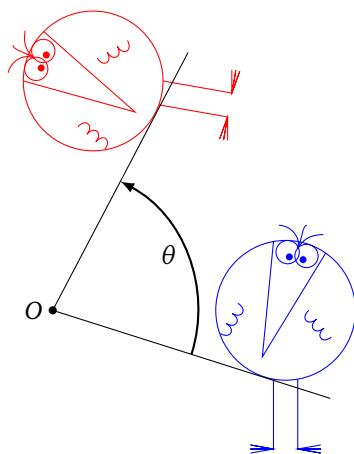


Il faut bien comprendre que le vecteur X est de longueur p mais le vecteur Y est de longueur n . Ainsi une matrice A correspond à une transformation : $X \mapsto Y = AX$.

Exemple.

La matrice de la rotation d'angle θ (centrée à l'origine) est :

$$R_\theta = \begin{pmatrix} \cos \theta & -\sin \theta \\ \sin \theta & \cos \theta \end{pmatrix}$$



Notons

$$X = \begin{pmatrix} x \\ y \end{pmatrix} \quad \text{et} \quad Y = R_\theta X$$

On calcule :

$$Y = \begin{pmatrix} x \cos \theta - y \sin \theta \\ x \sin \theta + y \cos \theta \end{pmatrix}$$

Ainsi en coordonnées cartésiennes la rotation d'angle θ s'écrit :

$$\begin{pmatrix} x \\ y \end{pmatrix} \mapsto \begin{pmatrix} x \cos \theta - y \sin \theta \\ x \sin \theta + y \cos \theta \end{pmatrix}$$

1.3. Produit de matrices

Le produit AB de deux matrices A et B est défini si et seulement si le nombre de colonnes de A est égal au nombre de lignes de B .

Définition (Produit de deux matrices).

Soient $A = (a_{ij})$ une matrice $n \times p$ et $B = (b_{ij})$ une matrice $p \times q$. Alors le **produit** $C = AB$ est une matrice $n \times q$ dont les coefficients c_{ij} sont définis par :

$$c_{ij} = \sum_{k=1}^p a_{ik} b_{kj}$$

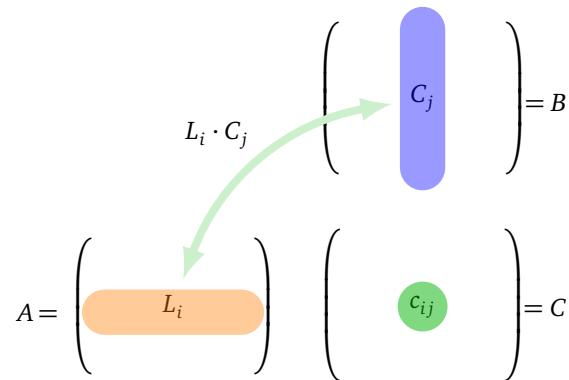
On peut écrire le coefficient de façon plus développée, à savoir :

$$c_{ij} = a_{i1}b_{1j} + a_{i2}b_{2j} + \cdots + a_{ik}b_{kj} + \cdots + a_{ip}b_{pj}.$$

Si on note L_i la i -ème ligne de la matrice A et C_j la j -ème colonne de la matrice B , alors

$$c_{ij} = L_i \cdot C_j.$$

Ainsi chaque coefficient de C est le résultat d'un produit scalaire entre une ligne de A avec une colonne de B .



Exemple.

$$A = \begin{pmatrix} 1 & 5 & -1 \\ 4 & 0 & 2 \end{pmatrix} \quad B = \begin{pmatrix} 2 & 0 \\ 1 & 3 \\ -1 & 1 \end{pmatrix}$$

On dispose d'abord le produit correctement (ci-dessous à gauche) : la matrice obtenue sera de taille 2×2 . Puis on calcule chacun des coefficients, en commençant par le premier coefficient $c_{11} = L_1 \cdot C_1 = 1 \times 2 + 5 \times 1 + (-1) \times (-1) = 8$ (au milieu), puis les autres (à droite).

$$\begin{array}{ccc}
 \begin{pmatrix} 2 & 0 \\ 1 & 3 \\ -1 & 1 \end{pmatrix} & \begin{pmatrix} 2 & 0 \\ 1 & 3 \\ -1 & 1 \end{pmatrix} & \begin{pmatrix} 2 & 0 \\ 1 & 3 \\ -1 & 1 \end{pmatrix} \\
 \begin{pmatrix} 1 & 5 & -1 \\ 4 & 0 & 2 \end{pmatrix} \begin{pmatrix} c_{11} & c_{12} \\ c_{21} & c_{22} \end{pmatrix} & \begin{pmatrix} 1 & 5 & -1 \\ 4 & 0 & 2 \end{pmatrix} \begin{pmatrix} 8 & c_{12} \\ c_{21} & c_{22} \end{pmatrix} & \begin{pmatrix} 1 & 5 & -1 \\ 4 & 0 & 2 \end{pmatrix} \begin{pmatrix} 8 & 14 \\ 6 & 2 \end{pmatrix} \\
 \text{Ainsi } AB = \begin{pmatrix} 8 & 14 \\ 6 & 2 \end{pmatrix}.
 \end{array}$$

1.4. $AB \neq BA$

Le produit de matrices n'est pas commutatif. Même dans le cas où AB et BA sont définis et de la même taille, on a en général $AB \neq BA$.

Exemple.

$$\begin{pmatrix} 1 & 4 \\ 2 & -3 \end{pmatrix} \begin{pmatrix} 0 & 2 \\ 2 & 5 \end{pmatrix} = \begin{pmatrix} 8 & 22 \\ -6 & -11 \end{pmatrix} \quad \text{mais} \quad \begin{pmatrix} 0 & 2 \\ 2 & 5 \end{pmatrix} \begin{pmatrix} 1 & 4 \\ 2 & -3 \end{pmatrix} = \begin{pmatrix} 4 & -6 \\ 12 & -7 \end{pmatrix}.$$

1.5. Opérations sur les matrices

Le produit de deux matrices est une opération compliquée mais la somme de deux matrices est une opération simple. Soient A et B deux matrices ayant la même taille $n \times p$. Leur **somme** $C = A + B$ est la matrice de taille $n \times p$ définie par

$$c_{ij} = a_{ij} + b_{ij}.$$

En d'autres termes, on somme coefficients par coefficients. On a bien $A + B = B + A$.

Exemple.

$$\text{Si } A = \begin{pmatrix} 5 & 8 \\ -1 & 2 \end{pmatrix} \quad \text{et} \quad B = \begin{pmatrix} 4 & 0 \\ 3 & 3 \end{pmatrix} \quad \text{alors} \quad A + B = \begin{pmatrix} 9 & 8 \\ 2 & 5 \end{pmatrix}.$$

Par contre souvenez-vous que AB et BA sont en général deux matrices différentes. Si on fait bien attention à l'ordre alors l'addition et la multiplication se comportent bien :

$$A(BC) = (AB)C \quad (\text{associativité})$$

$$A(B + C) = AB + AC \quad \text{et} \quad (B + C)A = BA + CA \quad (\text{distributivité})$$

En particulier pour un vecteur X et deux matrices A, B , on peut écrire $Y = ABX$ qui se calcule indifféremment par $Y = (AB)X$ ou $Y = A(BX)$.

Il est aussi facile de multiplier une matrice par un facteur $\alpha \in \mathbb{R}$: le produit d'une matrice $A = (a_{ij})$ de $M_{n,p}$ par un scalaire $\alpha \in \mathbb{R}$ est la matrice (αa_{ij}) formée en multipliant chaque coefficient de A par α . Elle est notée $\alpha \cdot A$ (ou simplement αA).

Exemple.

$$\text{Si } A = \begin{pmatrix} 4 & 3 & 2 \\ 2 & -1 & 0 \end{pmatrix} \quad \text{et} \quad \alpha = 2 \quad \text{alors} \quad \alpha A = \begin{pmatrix} 8 & 6 & 4 \\ 4 & -2 & 0 \end{pmatrix}.$$

La matrice $-A$ c'est $(-1)A$. Sur l'exemple ci-dessus on obtient : $-A = \begin{pmatrix} -4 & -3 & -2 \\ -2 & 1 & 0 \end{pmatrix}$.

1.6. Matrice d'un système linéaire

Un système d'équations linéaires peut s'écrire simplement à l'aide d'une matrice. Considérons le système linéaire ayant n équations et p inconnues (x_1, \dots, x_p) :

$$\left\{ \begin{array}{l} a_{11} x_1 + a_{12} x_2 + \cdots + a_{1p} x_p = b_1 \\ a_{21} x_1 + a_{22} x_2 + \cdots + a_{2p} x_p = b_2 \\ \vdots \\ a_{n1} x_1 + a_{n2} x_2 + \cdots + a_{np} x_p = b_n \end{array} \right.$$

Il s'écrit :

$$AX = B$$

où

$$\underbrace{\begin{pmatrix} a_{11} & \cdots & a_{1p} \\ a_{21} & \cdots & a_{2p} \\ \vdots & & \vdots \\ a_{n1} & \cdots & a_{np} \end{pmatrix}}_A \underbrace{\begin{pmatrix} x_1 \\ x_2 \\ \vdots \\ x_p \end{pmatrix}}_X = \underbrace{\begin{pmatrix} b_1 \\ b_2 \\ \vdots \\ b_n \end{pmatrix}}_B.$$

La matrice $A \in M_{n,p}$ s'appelle la matrice des coefficients du système ; $B \in M_{n,1}$ est le vecteur du second membre ; le vecteur $X \in M_{p,1}$ est une solution du système si et seulement si $AX = B$.

2. Vocabulaire

2.1. Matrices carrées

Si $n = p$ (même nombre de lignes que de colonnes), la matrice est dite **matrice carrée**. On note alors simplement M_n au lieu de $M_{n,n}$:

$$\begin{pmatrix} a_{11} & a_{12} & \cdots & a_{1i} & \cdots & a_{1n} \\ a_{21} & a_{22} & \cdots & a_{2i} & \cdots & a_{2n} \\ \vdots & \vdots & \ddots & \vdots & \ddots & \vdots \\ a_{i1} & a_{i2} & \cdots & a_{ii} & \cdots & a_{in} \\ \vdots & \vdots & \ddots & \vdots & \ddots & \vdots \\ a_{n1} & a_{n2} & \cdots & a_{ni} & \cdots & a_{nn} \end{pmatrix}$$

Les éléments $a_{11}, a_{22}, \dots, a_{nn}$ forment la **diagonale principale** de la matrice.

La matrice carrée suivante s'appelle la **matrice identité** :

$$I_n = \begin{pmatrix} 1 & 0 & \cdots & 0 \\ 0 & 1 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & 1 \end{pmatrix}$$

La matrice identité joue le rôle de l'unité pour la multiplication des matrices (comme la valeur 1 pour la multiplication des réels). Soit $A \in M_{n,p}$:

$$A \times I_p = A \quad \text{et} \quad I_n \times A = A$$

La matrice (de taille $n \times p$) dont tous les coefficients sont des zéros est appelée la **matrice nulle** et est notée $0_{n,p}$ ou plus simplement 0. Dans le calcul matriciel, la matrice nulle joue le rôle du nombre 0 pour les réels.

2.2. Matrices triangulaires, matrices diagonales

Soit A une matrice de taille $n \times n$. On dit que A est **triangulaire inférieure** si ses éléments au-dessus de la diagonale sont nuls, autrement dit :

$$i < j \implies a_{ij} = 0.$$

Une matrice triangulaire inférieure a la forme suivante :

$$\begin{pmatrix} a_{11} & 0 & \cdots & \cdots & 0 \\ a_{21} & a_{22} & \ddots & & \vdots \\ \vdots & \vdots & \ddots & \ddots & \vdots \\ \vdots & \vdots & & \ddots & 0 \\ a_{n1} & a_{n2} & \cdots & \cdots & a_{nn} \end{pmatrix}$$

On dit que A est **triangulaire supérieure** si ses éléments en-dessous de la diagonale sont nuls, elle a la forme suivante :

$$\begin{pmatrix} a_{11} & a_{12} & \cdots & \cdots & \cdots & a_{1n} \\ 0 & a_{22} & \cdots & \cdots & \cdots & a_{2n} \\ \vdots & \ddots & \ddots & & & \vdots \\ \vdots & & \ddots & \ddots & & \vdots \\ \vdots & & & \ddots & \ddots & \vdots \\ 0 & \cdots & \cdots & \cdots & 0 & a_{nn} \end{pmatrix}$$

Une matrice est **diagonale** lorsque elle est à la fois triangulaire inférieure et triangulaire supérieure. Autrement dit en dehors de la diagonale tous les coefficients sont nuls.

Exemple.

Une matrice triangulaire inférieure (à gauche), une matrice triangulaire supérieure (au centre), une matrice diagonale (à droite) :

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

2.3. Transposée

Soit A la matrice de taille $n \times p$

$$A = \begin{pmatrix} a_{11} & a_{12} & \cdots & a_{1p} \\ a_{21} & a_{22} & \cdots & a_{2p} \\ \vdots & \vdots & & \vdots \\ a_{n1} & a_{n2} & \cdots & a_{np} \end{pmatrix}.$$

La **matrice transposée** de A , notée A^T est la matrice de taille $p \times n$ définie par :

$$A^T = \begin{pmatrix} a_{11} & a_{21} & \dots & a_{n1} \\ a_{12} & a_{22} & \dots & a_{n2} \\ \vdots & \vdots & & \vdots \\ a_{1p} & a_{2p} & \dots & a_{np} \end{pmatrix}.$$

Autrement dit : le coefficient à la place (i, j) de A^T est a_{ji} . Ou encore la i -ème ligne de A devient la i -ème colonne de A^T (et réciproquement la j -ème colonne de A^T est la j -ème ligne de A).

Remarque : il existe aussi la notation ${}^t A$ pour la transposée de la matrice A .

En particulier la transposition transforme un vecteur ligne en un vecteur colonne et réciproquement :

$$\text{Si } X = \begin{pmatrix} x_1 & x_2 & \dots & x_n \end{pmatrix} \quad \text{alors} \quad X^T = \begin{pmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{pmatrix}.$$

Exemple.

$$\begin{aligned} \begin{pmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \\ 7 & 8 & 9 \end{pmatrix}^T &= \begin{pmatrix} 1 & 4 & 7 \\ 2 & 5 & 8 \\ 3 & 6 & 9 \end{pmatrix} \\ \begin{pmatrix} 1 & 8 \\ 0 & -5 \\ -2 & 3 \end{pmatrix}^T &= \begin{pmatrix} 1 & 0 & -2 \\ 8 & -5 & 3 \end{pmatrix} \quad (5 \ 3 \ -1)^T = \begin{pmatrix} 5 \\ 3 \\ -1 \end{pmatrix} \end{aligned}$$

L'opération de transposition obéit aux règles suivantes :

$$1. (A+B)^T = A^T + B^T$$

$$2. (\alpha A)^T = \alpha A^T$$

$$3. (A^T)^T = A$$

$$4. \boxed{(AB)^T = B^T A^T}$$

Notez bien l'inversion : $(AB)^T = B^T A^T$.

3. Transformations du plan

Voyons comment les matrices permettent de décrire beaucoup de transformations du plan. Dans les chapitres suivants « Transformations de l'espace » et « Rotations de l'espace » nous passerons à la dimension supérieure. Une **transformation affine** du plan est l'application $F : \mathbb{R}^2 \rightarrow \mathbb{R}^2$ définie par :

$$\begin{pmatrix} x \\ y \end{pmatrix} \mapsto \begin{pmatrix} a & b \\ c & d \end{pmatrix} \begin{pmatrix} x \\ y \end{pmatrix} + \begin{pmatrix} e \\ f \end{pmatrix},$$

où a, b, c, d, e, f sont des réels quelconques.

En d'autres termes, l'image d'un point (x, y) du plan est le point $F(x, y) = (x', y')$ avec

$$\begin{cases} x' = ax + by + e \\ y' = cx + dy + f \end{cases}.$$

En fait, une transformation affine F est la composée d'une transformation linéaire

$$\begin{pmatrix} x \\ y \end{pmatrix} \mapsto A \begin{pmatrix} x \\ y \end{pmatrix} \quad \text{où } A = \begin{pmatrix} a & b \\ c & d \end{pmatrix}$$

suivie d'une translation

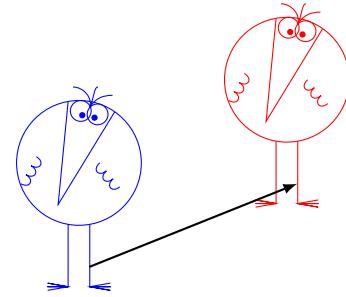
$$\begin{pmatrix} x \\ y \end{pmatrix} \mapsto \begin{pmatrix} x \\ y \end{pmatrix} + B \quad \text{où } B = \begin{pmatrix} e \\ f \end{pmatrix}.$$

Voici quelques transformations élémentaires.

Exemple.

1. Translation.

$$\begin{pmatrix} x \\ y \end{pmatrix} \mapsto \begin{pmatrix} x \\ y \end{pmatrix} + \begin{pmatrix} e \\ f \end{pmatrix}$$



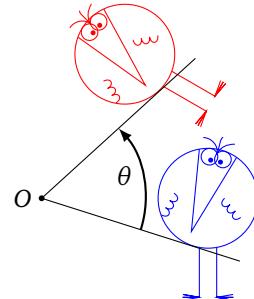
2. Rotation.

La **rotation** de centre l'origine et d'angle θ est l'application de \mathbb{R}^2 dans \mathbb{R}^2 définie par

$$\begin{pmatrix} x \\ y \end{pmatrix} \mapsto \begin{pmatrix} \cos \theta & -\sin \theta \\ \sin \theta & \cos \theta \end{pmatrix} \begin{pmatrix} x \\ y \end{pmatrix}$$

Autrement dit :

$$\begin{pmatrix} x \\ y \end{pmatrix} \mapsto \begin{pmatrix} x \cos \theta - y \sin \theta \\ x \sin \theta + y \cos \theta \end{pmatrix}$$



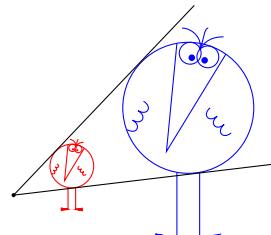
3. Homothétie.

L'**homothétie** de centre l'origine et de rapport $k \in \mathbb{R} \setminus \{0\}$ est l'application de \mathbb{R}^2 dans lui-même définie par :

$$\begin{pmatrix} x \\ y \end{pmatrix} \mapsto \begin{pmatrix} kx \\ ky \end{pmatrix}$$

En termes de matrice, l'écriture est la suivante :

$$\begin{pmatrix} x \\ y \end{pmatrix} \mapsto \begin{pmatrix} k & 0 \\ 0 & k \end{pmatrix} \begin{pmatrix} x \\ y \end{pmatrix}$$



4. Réflexion.

Nous commençons par regarder la réflexion par rapport à l'axe des abscisses : c'est l'application de \mathbb{R}^2 dans \mathbb{R}^2 définie par

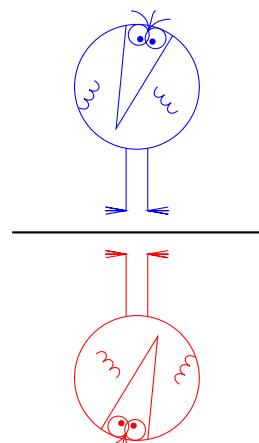
$$\begin{pmatrix} x \\ y \end{pmatrix} \mapsto \begin{pmatrix} x \\ -y \end{pmatrix}$$

En termes de matrice, l'écriture est la suivante :

$$\begin{pmatrix} x \\ y \end{pmatrix} \mapsto \begin{pmatrix} 1 & 0 \\ 0 & -1 \end{pmatrix} \begin{pmatrix} x \\ y \end{pmatrix}$$

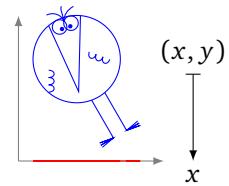
Plus généralement l'expression d'une réflexion par rapport à un axe passant par l'origine et faisant un angle $\frac{\theta}{2}$ avec l'axe des abscisses est

$$\begin{pmatrix} x \\ y \end{pmatrix} \mapsto \begin{pmatrix} \cos \theta & \sin \theta \\ \sin \theta & -\cos \theta \end{pmatrix} \begin{pmatrix} x \\ y \end{pmatrix}$$



5. Projection sur un axe.

Pour $M = \begin{pmatrix} 1 & 0 \\ 0 & 0 \end{pmatrix}$ et $e = f = 0$. La transformation affine est alors la projection $(x, y) \mapsto x$.

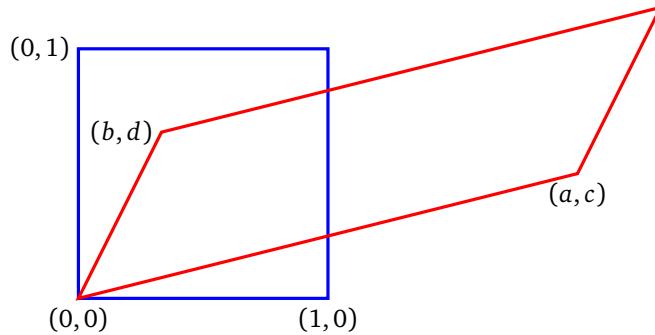


Exemple général

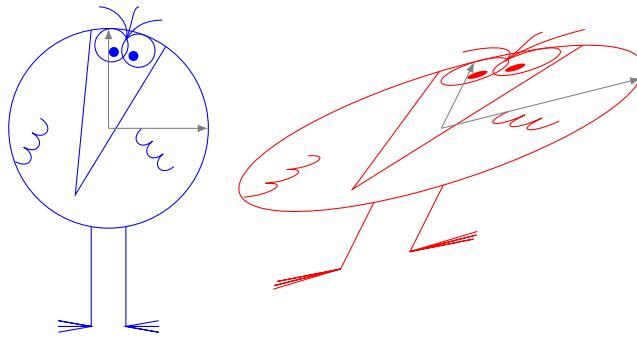
Oublions la translation et concentrons-nous sur l'application F définie par

$$\begin{pmatrix} x \\ y \end{pmatrix} \mapsto \begin{pmatrix} a & b \\ c & d \end{pmatrix} \begin{pmatrix} x \\ y \end{pmatrix}.$$

Nous avons $F(0, 0) = (0, 0)$, $F(1, 0) = (a, c)$, $F(0, 1) = (b, d)$. En termes de vecteurs, nous avons juste écrit que l'image du vecteur $\begin{pmatrix} 1 \\ 0 \end{pmatrix}$ était le premier vecteur colonne $\begin{pmatrix} a \\ c \end{pmatrix}$, alors que l'image du vecteur $\begin{pmatrix} 0 \\ 1 \end{pmatrix}$ est le second vecteur colonne $\begin{pmatrix} b \\ d \end{pmatrix}$. L'image du carré unitaire est donc un parallélogramme.



Remarquer que sur ce dessin, un côté vertical du carré est envoyé sur un petit côté du parallélogramme, et un côté horizontal sur un grand côté. Ni les longueurs ni les proportions ne sont conservées. Voici notre personnage et sa déformation :



Déterminant

Soit F une transformation affine de matrice $A = \begin{pmatrix} a & b \\ c & d \end{pmatrix}$. Le déterminant $\det(A) = ad - bc$ de cette matrice joue un rôle particulièrement important dans l'étude la transformation affine F .

Proposition 1.

La transformation F est bijective si et seulement si $\det(A) \neq 0$.

Proposition 2.

Si E est un ensemble dont l'aire vaut \mathcal{A} alors $F(E)$ est un ensemble dont l'aire vaut $|\det(A)| \times \mathcal{A}$.

4. Inverse

4.1. Inverse d'une matrice

Soit A une matrice carrée de taille $n \times n$. S'il existe une matrice carrée B de taille $n \times n$ telle que

$$AB = I \quad \text{et} \quad BA = I,$$

alors on dit que A est **inversible**. On appelle B l'**inverse de A** et on la note A^{-1} .

Proposition 3 (Cas des matrices 2×2).

Soit $A = \begin{pmatrix} a & b \\ c & d \end{pmatrix}$. Si $ad - bc \neq 0$, alors A est inversible et

$$A^{-1} = \frac{1}{ad - bc} \begin{pmatrix} d & -b \\ -c & a \end{pmatrix}$$

Nous n'expliquerons pas ici comment calculer l'inverse d'une matrice en général.

Proposition 4.

1. Soit A une matrice inversible. Alors A^{-1} est aussi inversible et on a :

$$(A^{-1})^{-1} = A$$

2. Soient A et B deux matrices inversibles de même taille. Alors AB est inversible et

$$(AB)^{-1} = B^{-1}A^{-1}$$

Pour l'inverse d'un produit il faut bien faire attention à l'inversion de l'ordre !

4.2. Matrices inversibles et systèmes linéaires

Considérons le cas où le nombre d'équations égale le nombre d'inconnues :

$$\underbrace{\begin{pmatrix} a_{11} & \dots & a_{1n} \\ a_{21} & \dots & a_{2n} \\ \vdots & & \vdots \\ a_{n1} & \dots & a_{nn} \end{pmatrix}}_A \underbrace{\begin{pmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{pmatrix}}_X = \underbrace{\begin{pmatrix} b_1 \\ b_2 \\ \vdots \\ b_n \end{pmatrix}}_B.$$

Alors $A \in M_n$ est une matrice carrée et B un vecteur de $M_{n,1}$. Pour tout second membre, nous pouvons utiliser les matrices pour trouver la solution du système linéaire.

Proposition 5.

Si la matrice A est inversible, alors la solution du système $AX = B$ est unique et est :

$$X = A^{-1}B$$

Les matrices sont un vaste sujet, ici nous avons présenté les notions indispensables à ce livre, mais vous trouverez plein d'autres propriétés (comme par exemple comment calculer l'inverse) dans n'importe quel ouvrage d'algèbre linéaire, par exemple le tome Algèbre d'Exo7 dont certains paragraphes de ce chapitre sont tirés.

Transformations de l'espace

Nous étudions les transformations affines usuelles de l'espace : translations, homothéties, réflexions... à l'aide des vecteurs et matrices. Nous décrivons les formules de changement de base et introduisons les coordonnées homogènes.

1. Transformations affines

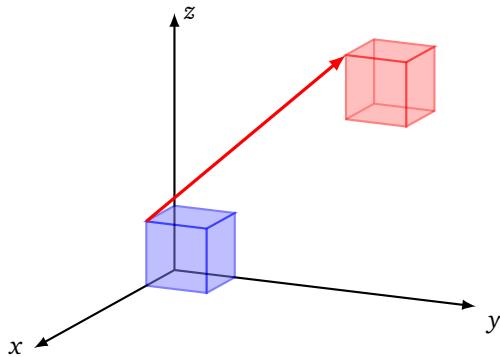
1.1. Translations

Une **translation** de vecteur (a, b, c) est la fonction de \mathbb{R}^3 dans \mathbb{R}^3 définie par

$$\begin{pmatrix} x \\ y \\ z \end{pmatrix} \mapsto \begin{pmatrix} x' \\ y' \\ z' \end{pmatrix} = \begin{pmatrix} x \\ y \\ z \end{pmatrix} + \begin{pmatrix} a \\ b \\ c \end{pmatrix} = \begin{pmatrix} x+a \\ y+b \\ z+c \end{pmatrix}$$

Si on note $T = \begin{pmatrix} a \\ b \\ c \end{pmatrix}$ alors, l'image Y d'un point $X = \begin{pmatrix} x \\ y \\ z \end{pmatrix}$ est :

$$Y = X + T.$$



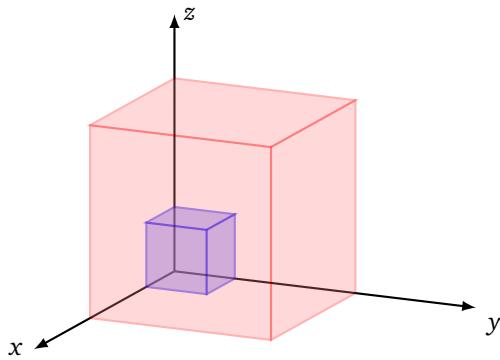
1.2. Homothéties

Une **homothétie** centrée à l'origine et de rapport k est l'application $\mathbb{R}^3 \rightarrow \mathbb{R}^3$, $X \mapsto Y$ définie par

$$Y = kX$$

Autrement dit $x' = kx$, $y' = ky$, $z' = kz$. Nous préférons écrire les transformations en termes de vecteurs et matrices :

$$Y = AX \quad \text{avec} \quad A = \begin{pmatrix} k & 0 & 0 \\ 0 & k & 0 \\ 0 & 0 & k \end{pmatrix}.$$

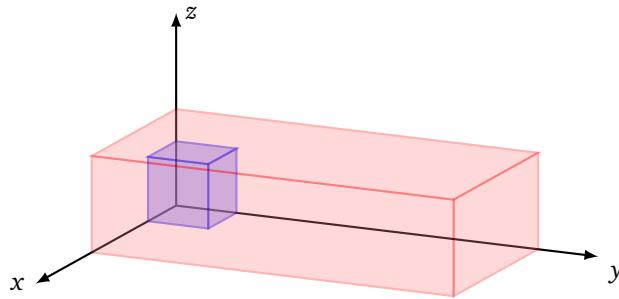


Si on souhaite une homothétie de rapport k centrée en un point quelconque $X_0 = \begin{pmatrix} x_0 \\ y_0 \\ z_0 \end{pmatrix}$, alors on applique la formule :

$$Y = A(X - X_0) + X_0.$$

Pour déformer l'espace avec des rapports différents selon chaque axe (ce n'est plus une homothétie), on utiliserait la matrice :

$$A = \begin{pmatrix} k_x & 0 & 0 \\ 0 & k_y & 0 \\ 0 & 0 & k_z \end{pmatrix}.$$

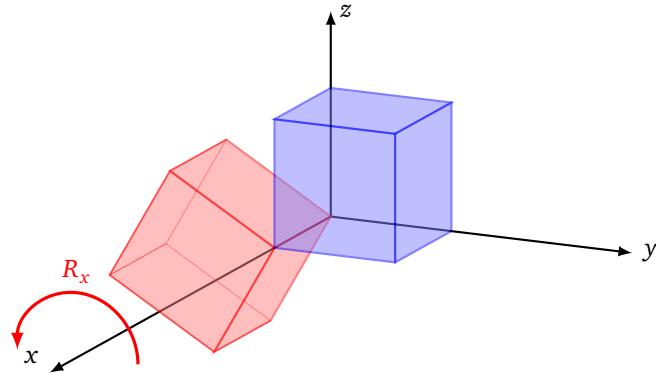


1.3. Rotations

Les rotations seront étudiées en détail dans le chapitre « Rotations de l'espace ».

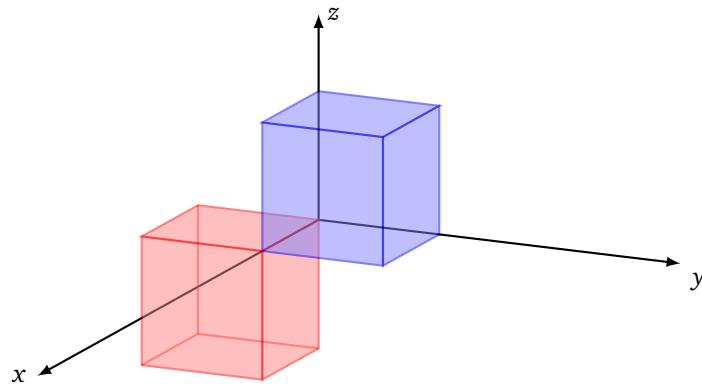
Par exemple la rotation d'axe (Ox) et d'angle θ est la transformation $Y = R_x(\theta)X$ avec

$$R_x(\theta) = \begin{pmatrix} 1 & 0 & 0 \\ 0 & \cos(\theta) & -\sin(\theta) \\ 0 & \sin(\theta) & \cos(\theta) \end{pmatrix}.$$



Une rotation d'angle π (180°) s'appelle un **retournement**. Le retournement d'axe (Ox) a pour matrice :

$$R_x(\pi) = \begin{pmatrix} 1 & 0 & 0 \\ 0 & -1 & 0 \\ 0 & 0 & -1 \end{pmatrix}.$$



Voici les matrices de rotations autour de l'axe (Oy) et de l'axe (Oz) :

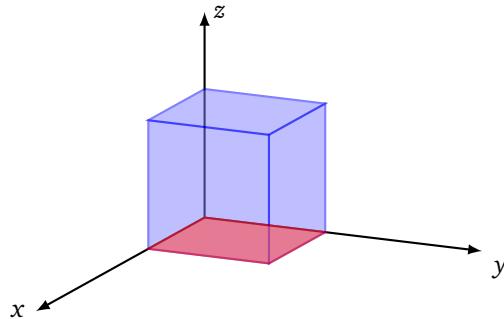
$$R_y(\theta) = \begin{pmatrix} \cos(\theta) & 0 & \sin(\theta) \\ 0 & 1 & 0 \\ -\sin(\theta) & 0 & \cos(\theta) \end{pmatrix} \quad R_z(\theta) = \begin{pmatrix} \cos(\theta) & -\sin(\theta) & 0 \\ \sin(\theta) & \cos(\theta) & 0 \\ 0 & 0 & 1 \end{pmatrix}.$$

1.4. Projections

Les projections seront étudiées en détail dans le chapitre « Perspective ».

Par exemple la **projection orthogonale** sur le plan (Oxy) est la transformation $Y = AX$ avec

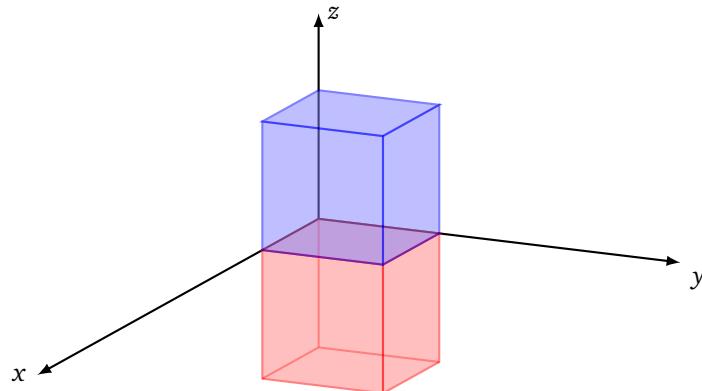
$$A = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \end{pmatrix}.$$



1.5. Réflexions

La **réflexion orthogonale** par rapport au plan (Oxy) est la transformation $Y = AX$ avec

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



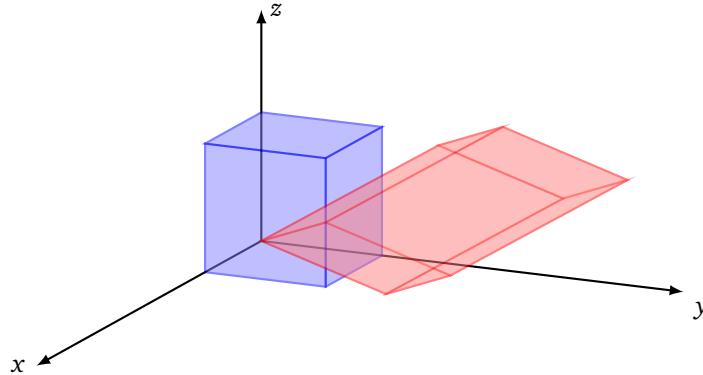
Plus généralement si A est la matrice d'une projection sur un sous-espace, alors $B = 2A - I$ est la matrice de la réflexion par rapport à ce même sous-espace.

1.6. Matrice quelconque

De façon générale une **transformation vectorielle** est l'application $F : \mathbb{R}^3 \rightarrow \mathbb{R}^3, X \mapsto Y$ où :

$$Y = AX \quad \text{avec } A \in M_3(\mathbb{R}).$$

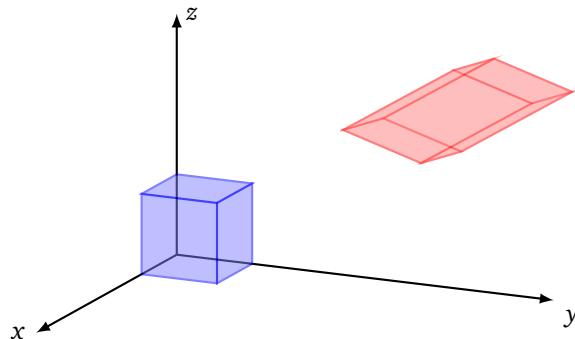
On appelle aussi la fonction F une **application linéaire**.



Une **transformation affine** est une transformation vectorielle, suivie d'une translation :

$$Y = AX + T \quad \text{avec } A \in M_3(\mathbb{R}), T \in M_{3,1}(\mathbb{R}).$$

Une transformation vectorielle envoie toujours l'origine sur l'origine, à la différence d'une transformation affine.



Soit $F : \mathbb{R}^3 \rightarrow \mathbb{R}^3$ une transformation affine ou vectorielle. Notons A la matrice de cette transformation. Le déterminant $\det(A)$ de cette matrice est important dans l'étude de la transformation F .

Proposition 1.

La transformation F est bijective si et seulement si $\det(A) \neq 0$.

Proposition 2.

Si E est un ensemble dont le volume est \mathcal{V} alors $F(E)$ est un ensemble dont le volume est $|\det(A)| \times \mathcal{V}$.

Rappelons que si $A \in M_3(\mathbb{R})$ est une matrice 3×3 :

$$A = \begin{pmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ a_{31} & a_{32} & a_{33} \end{pmatrix}$$

alors le déterminant se calcule selon la formule :

$$\det(A) = a_{11}a_{22}a_{33} + a_{12}a_{23}a_{31} + a_{13}a_{21}a_{32} - a_{31}a_{22}a_{13} - a_{32}a_{23}a_{11} - a_{33}a_{21}a_{12}.$$

Les matrices permettent d'effectuer facilement la composition des transformations vectorielles. Si F a pour matrice A et G a pour matrice B , alors la transformation $F \circ G$ (l'action de G suivie de celle de F) a pour matrice le produit AB . C'est-à-dire : $F \circ G : X \mapsto Y = (AB)X$. On rappelle que l'ordre a une importance, les matrices AB et BA sont en général distinctes, autrement dit, appliquer F puis G n'est pas la même chose qu'appliquer G puis F .

2. Changement de repère

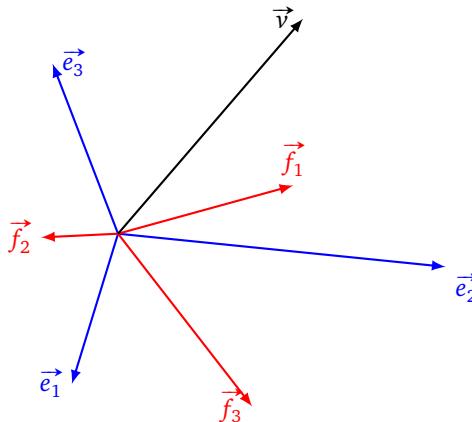
2.1. Changement de coordonnées

Soit $\mathcal{B} = (\vec{e}_1, \vec{e}_2, \vec{e}_3)$ une base de \mathbb{R}^3 . Considérons un vecteur \vec{v} de \mathbb{R}^3 et notons X les coordonnées de \vec{v} dans la base \mathcal{B} , c'est-à-dire :

$$\vec{v} = x\vec{e}_1 + y\vec{e}_2 + z\vec{e}_3 \quad \text{et} \quad X = \begin{pmatrix} x \\ y \\ z \end{pmatrix}.$$

Fixons maintenant une seconde base $\mathcal{B}' = (\vec{f}_1, \vec{f}_2, \vec{f}_3)$ de \mathbb{R}^3 . Le même vecteur \vec{v} n'a pas les mêmes coordonnées dans cette nouvelle base. Notons X' les coordonnées de \vec{v} dans cette base \mathcal{B}' , c'est-à-dire :

$$\vec{v} = x'\vec{f}_1 + y'\vec{f}_2 + z'\vec{f}_3 \quad \text{et} \quad X' = \begin{pmatrix} x' \\ y' \\ z' \end{pmatrix}.$$



Quel est le lien entre X et X' ?

La **matrice de passage** P de la base \mathcal{B} vers la base \mathcal{B}' est la matrice carrée de taille 3×3 dont la j -ème colonne est formée des coordonnées du j -ème vecteur de la base \mathcal{B}' , par rapport à la base \mathcal{B} .

On résume en :

La matrice de passage P contient – en colonnes – les coordonnées des vecteurs de la nouvelle base \mathcal{B}' exprimés dans l'ancienne base \mathcal{B} .

Voici le lien entre les coordonnées dans l'ancienne et la nouvelle base :

Proposition 3.

$$X = PX'$$

Notez bien l'ordre ! La formule permet de calculer les coordonnées dans la base de départ à partir de celle de la base d'arrivée. Mais en général on veut l'opération inverse. Pour cela on utilise simplement la relation $X' = P^{-1}X$ qui donne les coordonnées dans la nouvelle base à partir des coordonnées dans l'ancienne base.

Exemple.

Considérons \mathbb{R}^3 muni de sa base canonique \mathcal{B} , mais aussi d'une autre base \mathcal{B}' avec :

$$\mathcal{B} = \left(\begin{pmatrix} 1 \\ 0 \\ 0 \end{pmatrix}, \begin{pmatrix} 0 \\ 1 \\ 0 \end{pmatrix}, \begin{pmatrix} 0 \\ 0 \\ 1 \end{pmatrix} \right) \quad \text{et} \quad \mathcal{B}' = \left(\begin{pmatrix} 1 \\ 1 \\ 0 \end{pmatrix}, \begin{pmatrix} 0 \\ 2 \\ 1 \end{pmatrix}, \begin{pmatrix} 1 \\ 2 \\ 3 \end{pmatrix} \right).$$

Quelle est la matrice de passage de \mathcal{B} vers \mathcal{B}' ?

Comme la base de départ est la base canonique alors dans ce cas la matrice de passage est simplement la matrice dont les colonnes sont les vecteurs de la base \mathcal{B}' , ainsi :

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

Nous aurons besoin de calculer son inverse. Après calculs :

$$P^{-1} = \frac{1}{5} \begin{pmatrix} 4 & 1 & -2 \\ -3 & 3 & -1 \\ 1 & -1 & 2 \end{pmatrix}$$

Considérons un vecteur dont les coordonnées dans la base \mathcal{B} sont :

$$X = \begin{pmatrix} 5 \\ 6 \\ 8 \end{pmatrix}$$

Quelles sont les coordonnées X' de ce même vecteur dans la base \mathcal{B}' ? Comme $X = PX'$ alors $X' = P^{-1}X$, ainsi :

$$X' = P^{-1}X = \frac{1}{5} \begin{pmatrix} 4 & 1 & -2 \\ -3 & 3 & -1 \\ 1 & -1 & 2 \end{pmatrix} \begin{pmatrix} 5 \\ 6 \\ 8 \end{pmatrix} = \begin{pmatrix} 2 \\ -1 \\ 3 \end{pmatrix}.$$

2.2. Changement de base pour les fonctions

Soit $F : \mathbb{R}^3 \rightarrow \mathbb{R}^3$ une application linéaire (c'est-à-dire une transformation vectorielle). Notons A la matrice de F dans la base \mathcal{B} . Ainsi si $f(\vec{v}) = \vec{w}$ et que \vec{v} a pour coordonnées X et \vec{w} a pour coordonnées Y (toujours dans la même base \mathcal{B}) alors

$$Y = AX$$

Très souvent, la base choisie est la base canonique et alors on définit une application linéaire par sa matrice. Mais la relation est en général plus subtile :

$$(une matrice + le choix d'une base) \longleftrightarrow \text{une application linéaire}$$

Donc dans une autre base \mathcal{B}' , la matrice de F est différente : notons B cette matrice. Comment exprimer B en fonction de A ?

La formule de changement de base pour une application linéaire est :

Proposition 4.

$$B = P^{-1}AP$$

Comme auparavant, la matrice P est la matrice de passage de la base \mathcal{B} à la base \mathcal{B}' .

Exemple.

Reprendons les deux bases de \mathbb{R}^3 de l'exemple du paragraphe 2.1 :

$$\mathcal{B} = \left(\begin{pmatrix} 1 \\ 0 \\ 0 \end{pmatrix}, \begin{pmatrix} 0 \\ 1 \\ 0 \end{pmatrix}, \begin{pmatrix} 0 \\ 0 \\ 1 \end{pmatrix} \right) \quad \text{et} \quad \mathcal{B}' = \left(\begin{pmatrix} 1 \\ 1 \\ 0 \end{pmatrix}, \begin{pmatrix} 0 \\ 2 \\ 1 \end{pmatrix}, \begin{pmatrix} 1 \\ 2 \\ 3 \end{pmatrix} \right).$$

Considérons la rotation F d'axe (Oz) et d'angle $\frac{\pi}{2}$. Sa matrice dans la base \mathcal{B} est

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

Autrement dit, dans la base \mathcal{B} un vecteur de coordonnées X s'envoie sur le vecteur de coordonnées $Y = AX$. Quelle est la matrice B de cette même rotation, mais dans la base \mathcal{B}' ? On cherche la matrice B telle que dans la base \mathcal{B}' cette fois un vecteur de coordonnées X' s'envoie sur les coordonnées $Y' = BX'$. La formule de changement de base pour les matrices est $B = P^{-1}AP$, donc :

$$B = P^{-1}AP = \frac{1}{5} \begin{pmatrix} 4 & 1 & -2 \\ -3 & 3 & -1 \\ 1 & -1 & 2 \end{pmatrix} \begin{pmatrix} 0 & -1 & 0 \\ 1 & 0 & 0 \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} 1 & 0 & 1 \\ 1 & 2 & 2 \\ 0 & 1 & 3 \end{pmatrix} = \frac{1}{5} \begin{pmatrix} -3 & -10 & -13 \\ 6 & 5 & 6 \\ -2 & 0 & 3 \end{pmatrix}.$$

Par exemple, pour un vecteur ayant pour coordonnées $X' = \begin{pmatrix} 5 \\ 0 \\ 10 \end{pmatrix}$ dans la base \mathcal{B}' , alors son image par la rotation F aura pour coordonnées $Y' = BX' = \begin{pmatrix} -29 \\ 18 \\ 4 \end{pmatrix}$ (toujours dans la base \mathcal{B}').

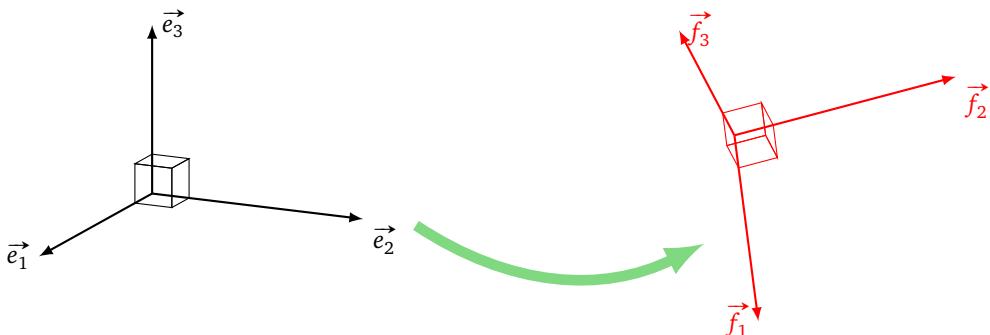
2.3. Changement de base orthonormée

On rappelle qu'une base \mathcal{B} est **orthonormale** si chaque vecteur est unitaire et si deux vecteurs distincts sont orthogonaux.

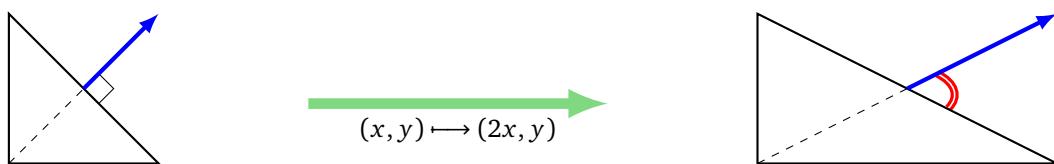
Une matrice A est **orthogonale** si $A^T A = I$, autrement dit si $A^{-1} = A^T$. De façon équivalente, une matrice A est orthogonale si ses vecteurs colonnes forment une base orthonormale. On note $O(3)$ l'ensemble des matrices orthogonales de taille 3×3 .

Proposition 5.

Si \mathcal{B} et \mathcal{B}' sont deux bases orthonormales alors la matrice de passage P de \mathcal{B} à \mathcal{B}' est une matrice orthogonale.



Il faut aussi prendre garde qu'une transformation vectorielle ne préserve en général pas l'orthogonalité (même si c'est vrai pour les homothéties, les rotations, les symétries orthogonales).



Proposition 6.

Soit A la matrice d'une application linéaire F . Si A est une matrice orthogonale alors F préserve le produit scalaire, c'est-à-dire $F(\vec{u}) \cdot F(\vec{v}) = \vec{u} \cdot \vec{v}$. En particulier F préserve les angles et les longueurs ; ainsi F préserve l'orthogonalité et envoie une base orthonormale sur une base orthonormale.

Conséquence : si dans une base orthonormée F a pour matrice la matrice orthogonale A , alors dans une autre base orthonormée F a pour matrice B qui est aussi orthogonale (c'est $B = P^{-1}AP$ avec A et P orthogonales).

Exercice.

On considère la matrice

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

1. Vérifier que $A^{-1} = A^T$, en déduire que A est une matrice orthogonale.
2. Montrer que les vecteurs de coordonnées $X_1 = \begin{pmatrix} 2 \\ 3 \\ 7 \end{pmatrix}$ et $X_2 = \begin{pmatrix} 2 \\ 1 \\ -1 \end{pmatrix}$ sont orthogonaux.
3. Calculer les coordonnées $Y_1 = AX_1$ de l'image de X_1 par A . Idem pour $Y_2 = AX_2$. Vérifier que Y_1 et Y_2 sont encore des vecteurs orthogonaux.

3. Coordonnées homogènes

3.1. Motivation

Il y a plusieurs inconvénients à la description des transformations vues lors des sections précédentes : la plupart des transformations étudiées jusqu'ici étaient des transformations vectorielles (où l'origine s'envoie sur l'origine) et les translations sont effectuées à part afin d'obtenir une transformation affine. D'autre part les ordinateurs savent multiplier très rapidement des matrices (pour composer les applications linéaires), mais les translations requièrent un traitement à part (une addition). Pourrait-on unifier la situation ? Un autre problème est de manipuler des objets à l'infini. Par exemple, pour un éclairage, il faut différencier un éclairage issu d'un point, d'un éclairage « à l'infini » comme le Soleil. Encore une fois : comment unifier cette situation ?

Ces deux problèmes sont réglés par les coordonnées homogènes. Il s'agit d'ajouter une coordonnée supplémentaire, ainsi un point de l'espace est codé avec 4 nombres réels et une transformation qui inclut une translation est codée à l'aide d'une matrice 4×4 . Les points à l'infini sont les points dont la dernière coordonnée est nulle.

Pour mieux comprendre et pouvoir faire des dessins on commence par expliquer les coordonnées homogènes du plan.

3.2. Coordonnées homogènes du plan

Définition

On note $(x : y : w)$ les **coordonnées homogènes** du plan où x, y, w sont des réels (pas tous les trois nuls en même temps). Ces coordonnées sont définies à un facteur près, c'est-à-dire que :

$$(x : y : w) = (\lambda x : \lambda y : \lambda w) \quad \text{pour tout } \lambda \in \mathbb{R}^*$$

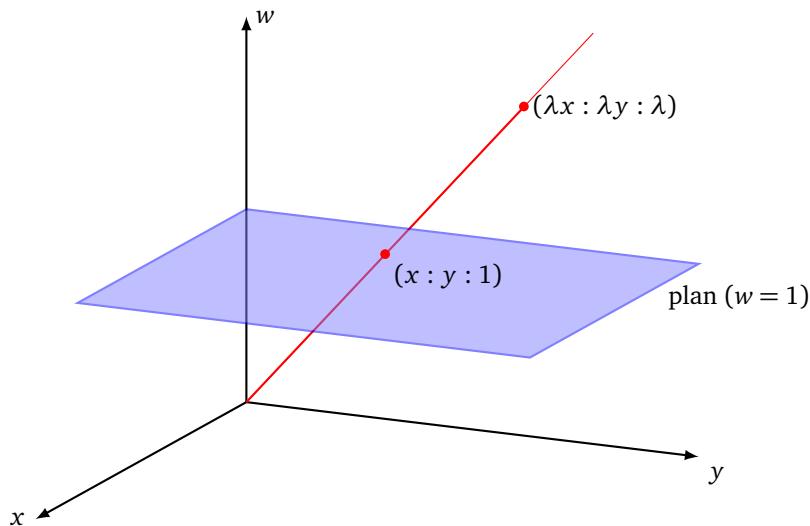
Par exemple $(2 : -1 : 1) = (4 : -2 : 2) = (-6 : 3 : -3)$ et $(2 : 3 : 0) = (4 : 6 : 0)$. Attention, le point « $(0 : 0 : 0)$ » n'existe pas.

On appelle **plan projectif**, noté \mathbb{RP}^2 , l'ensemble de ces triplets $(x : y : w)$.

- Si $(x, y) \in \mathbb{R}^2$ est un point du plan alors on lui associe les coordonnées homogènes $(x : y : 1)$.
- Réciproquement à $(x : y : w)$ avec $w \neq 0$, on lui associe le point $(x/w, y/w)$. Noter que si $w \neq 0$ on a $(x : y : w) = (x/w : y/w : 1)$.
- Si (v_x, v_y) est un vecteur du plan, on lui associe les coordonnées homogènes $(v_x : v_y : 0)$, aussi appelé « point à l'infini ». Réciproquement à $(v_x : v_y : 0)$, on associe le vecteur (ou la direction) (v_x, v_y) .

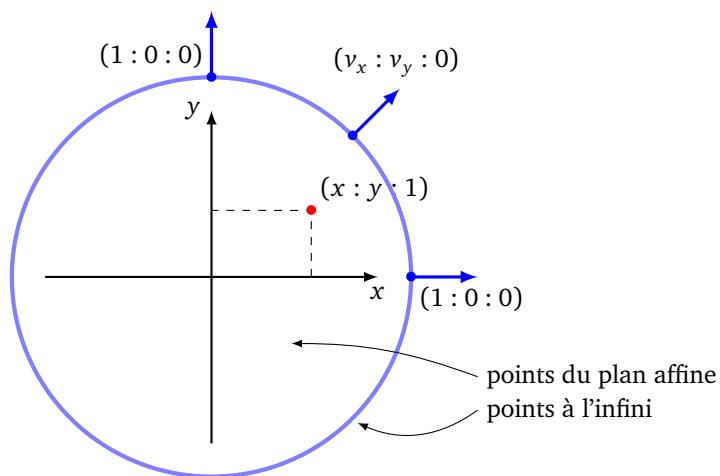
Pour décrire le plan projectif d'un point de vue géométrique, on part de l'espace \mathbb{R}^3 et on identifie les points qui sont situés sur une même droite passant par l'origine (car $(x : y : w) = (\lambda x : \lambda y : \lambda w)$).

L'identification $(x : y : 1)$ avec le point (x, y) correspond à intersecer une droite vectorielle de l'espace avec le plan d'équation $(w = 1)$.



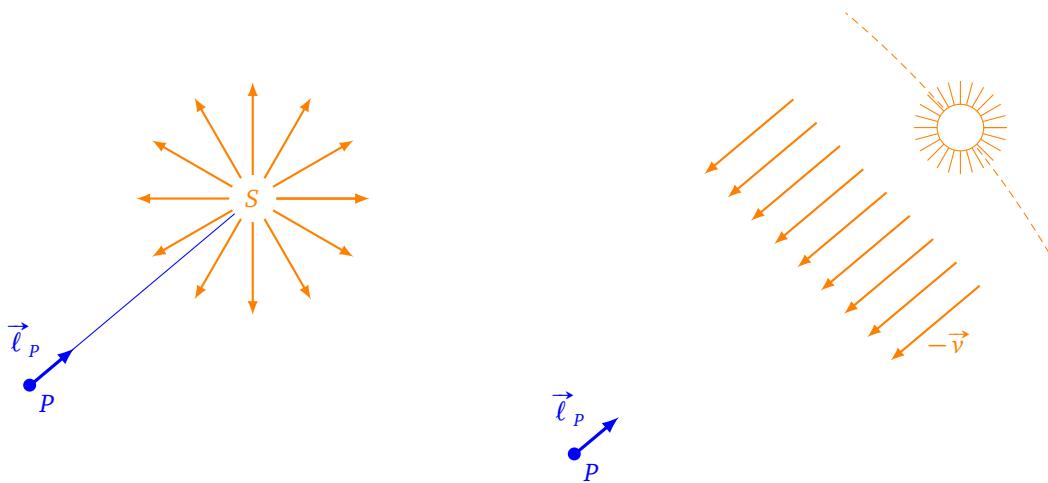
Points à l'infini

On peut se représenter le plan projectif ainsi : une partie affine correspondant aux points de coordonnées homogènes $(x : y : 1)$ et un ensemble de points à l'infini de coordonnées homogènes $(v_x : v_y : 0)$. Un point à l'infini $(v_x : v_y : 0)$ correspond à une direction $\vec{v} = (v_x, v_y)$.



Voyons maintenant comment uniformiser la position d'un éclairage. La source d'un éclairage est définie par un point $S \in \mathbb{RP}^2$ de coordonnées homogènes $(x_S : y_S : w_S)$ avec $w_S = 0$ ou bien $w_S = 1$.

- **Lumière ponctuelle.** $S = (x_S : y_S : 1)$. Dans ce cas la source lumineuse est en position (x_S, y_S) . Si $P(x, y)$ est un point du plan, alors un vecteur unitaire dirigé vers la source lumineuse est $\ell_P = \frac{\vec{PS}}{\|\vec{PS}\|}$.
- **Lumière directionnelle.** $S = (x_S : y_S : 0)$. Dans ce cas la source lumineuse est « à l'infini » et est caractérisée par la direction opposée à $\vec{v} = (x_S, y_S)$. Pour n'importe quel point P du plan, $\ell = \frac{\vec{v}}{\|\vec{v}\|}$ est un vecteur unitaire dirigé parallèlement aux rayons lumineux.



Transformation

Soit $F : \mathbb{R}^2 \rightarrow \mathbb{R}^2$ une transformation affine du plan :

$$\begin{pmatrix} x \\ y \end{pmatrix} \mapsto \begin{pmatrix} a & b \\ c & d \end{pmatrix} \begin{pmatrix} x \\ y \end{pmatrix} + \begin{pmatrix} e \\ f \end{pmatrix},$$

où a, b, c, d, e, f sont des réels quelconques.

En d'autres termes, l'image d'un point (x, y) du plan est le point $F(x, y) = (x', y')$ avec

$$\begin{cases} x' = ax + by + e \\ y' = cx + dy + f \end{cases}.$$

Si on note :

$$A = \begin{pmatrix} a & b \\ c & d \end{pmatrix} \quad T = \begin{pmatrix} e \\ f \end{pmatrix} \quad X = \begin{pmatrix} x \\ y \end{pmatrix}$$

alors $F(X) = AX + T$.

Problème de la composition

Composer deux transformations vectorielles est simple : si F a pour matrice A et G a pour matrice B alors $F \circ G$ a pour matrice AB . La composition correspond simplement au produit de matrices.

Faisons maintenant le calcul avec des transformations affines $F : X \mapsto AX + T$ et $G : X \mapsto BX + S$:

$$F \circ G(X) = F(G(X)) = F(BX + S) = A(BX + S) + T = ABX + (AS + T).$$

La formule n'est donc pas simple et se compliquerait encore si ajoutait des compositions.

Calculons l'action de la transformation $F : X \mapsto AX + T$ en coordonnées homogènes.

$$\text{À } X = \begin{pmatrix} x \\ y \end{pmatrix} \quad \text{on associe } X_h = \begin{pmatrix} x \\ y \\ 1 \end{pmatrix}.$$

Et à la transformation affine (de matrice A et translation T) on associe la matrice :

$$A_h = \begin{pmatrix} a & b & e \\ c & d & f \\ 0 & 0 & 1 \end{pmatrix}$$

Vérifions que $F(X_h) = A_h X_h$ (en identifiant un point $P_h = \begin{pmatrix} x \\ y \\ 1 \end{pmatrix}$ avec $(x : y : 1)$ et (x, y)) :

$$A_h X_h = \begin{pmatrix} a & b & e \\ c & d & f \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} x \\ y \\ 1 \end{pmatrix} = \begin{pmatrix} ax + by + e \\ cx + dy + f \\ 1 \end{pmatrix} = F(X_h).$$

Ainsi en coordonnées homogènes, une transformation affine du plan correspond à la multiplication par une matrice 3×3 .

Si $G : X \mapsto BX + S$ est une autre transformation affine et que l'on note B_h la matrice 3×3 associée, alors

$$F \circ G(X_h) = F(G(X)) = F(B_h X_h) = A_h (B_h X_h) = (A_h B_h) X_h.$$

Ainsi, en coordonnées homogènes, la matrice associée à $F \circ G$ est naturellement le produit $A_h B_h$.

3.3. Coordonnées homogènes de l'espace

Ajoutons une dimension supplémentaire afin de définir les coordonnées homogènes dans l'espace.

Coordonnées homogènes

On note $(x : y : z : w)$ les **coordonnées homogènes** de l'espace, où x, y, z et w sont des réels, pas tous les quatre nuls en même temps. Ces coordonnées sont définies à un facteur multiplicatif près, c'est-à-dire que :

$$(x : y : z : w) = (\lambda x : \lambda y : \lambda z : \lambda w) \quad \text{pour tout } \lambda \in \mathbb{R}^*$$

L'ensemble de ces éléments $(x : y : z : w)$, à équivalence près, s'appelle l'**espace projectif**, noté \mathbb{RP}^3 .

Les coordonnées classiques correspondent aux coordonnées homogènes lorsque $w = 1$. Plus précisément :

- Si $X = (x, y, z) \in \mathbb{R}^3$ est un point de l'espace alors on lui associe les coordonnées homogènes $X_h = (x : y : z : 1)$.
- Réciproquement à $(x : y : z : w)$ avec $w \neq 0$, on lui associe le point $(x/w, y/w, z/w)$. Noter que si $w \neq 0$ on a $(x : y : z : w) = (x/w : y/w : z/w : 1)$.
- Si (v_x, v_y, v_z) est un vecteur, on lui associe les coordonnées homogènes $(v_x : v_y : v_z : 0)$, aussi appelé « un point à l'infini ». Réciproquement à $(v_x : v_y : v_z : 0)$, on associe le vecteur (ou la direction) (v_x, v_y, v_z) .

Exemple.

1. Le point $\bar{A} \in \mathbb{RP}^3$ de coordonnées homogènes $(-3 : 0 : 2 : 1)$ a aussi pour coordonnées homogènes $(-6 : 0 : 4 : 2)$. Le point $A \in \mathbb{R}^3$ correspondant est $(-3, 0, 2)$.
2. Le point $\bar{B} \in \mathbb{RP}^3$ de coordonnées $(2 : 3 : -2 : 1)$ est associé à $B \in \mathbb{R}^3$ de coordonnées $(2, 3, -2)$.
3. Lorsque les coordonnées homogènes sont normalisées avec $w = 1$ on peut soustraire deux points pour obtenir un vecteur :

$$\bar{B} - \bar{A} = (2 : 3 : -2 : 1) - (-3 : 0 : 2 : 1) = (5 : 3 : -4 : 0)$$

qui est un point à l'infini et correspond bien aux coordonnées homogènes du vecteur \vec{AB} .

Il est difficile de visualiser l'espace projectif. Un premier point de vue est de partir de l'espace \mathbb{R}^4 (à quatre dimensions) et d'identifier les points qui sont situés sur une même droite vectorielle. Une autre vision est de considérer que \mathbb{RP}^3 correspond à l'ensemble des points de \mathbb{R}^3 auxquels on rajoute des points à l'infini (qui sont en fait en bijection avec le plan projectif \mathbb{RP}^2).

Transformations homogènes

Soit $F : \mathbb{R}^3 \rightarrow \mathbb{R}^3$ une transformation affine de l'espace définie par :

$$F(X) = AX + T$$

où $A \in M_3(\mathbb{R})$ est une matrice 3×3 et $T \in M_{3,1}(\mathbb{R})$ est un vecteur colonne de taille 3 correspondant à la translation.

On note :

$$A_h = \left(\begin{array}{c|c} A & T \\ \hline 0 & 1 \end{array} \right) \in M_4(\mathbb{R})$$

Autrement dit, si

$$A = \begin{pmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ a_{31} & a_{32} & a_{33} \end{pmatrix} \quad \text{et} \quad T = \begin{pmatrix} t_1 \\ t_2 \\ t_3 \end{pmatrix} \quad \text{alors} \quad A_h = \begin{pmatrix} a_{11} & a_{12} & a_{13} & t_1 \\ a_{21} & a_{22} & a_{23} & t_2 \\ a_{31} & a_{32} & a_{33} & t_3 \\ \hline 0 & 0 & 0 & 1 \end{pmatrix}.$$

Vérifions que $F(X_h) = A_h X_h$:

$$A_h X_h = \begin{pmatrix} a_{11} & a_{12} & a_{13} & t_1 \\ a_{21} & a_{22} & a_{23} & t_2 \\ a_{31} & a_{32} & a_{33} & t_3 \\ \hline 0 & 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} x \\ y \\ z \\ 1 \end{pmatrix} = \begin{pmatrix} a_{11}x + a_{12}y + a_{13}z + t_1 \\ a_{21}x + a_{22}y + a_{23}z + t_2 \\ a_{31}x + a_{32}y + a_{33}z + t_3 \\ 1 \end{pmatrix} = F(X_h).$$

Ainsi, en coordonnées homogènes, une transformation affine de l'espace correspond à la multiplication par une matrice 4×4 .

Exemple.

Soit A_h la matrice homogène d'une transformation $F(X) = AX + T$ définie par :

$$\left(\begin{array}{ccc|c} 1 & 0 & -1 & 1 \\ 2 & 1 & 0 & 2 \\ -2 & 1 & 1 & 3 \\ \hline 0 & 0 & 0 & 1 \end{array} \right).$$

1. Calculons l'image d'un point de coordonnées $X = (4, -2, 3)$ par la transformation F .

Ses coordonnées homogènes sont $X_h = (4 : -2 : 3 : 1)$. Alors :

$$Y_h = A_h X_h = \left(\begin{array}{cccc} 1 & 0 & -1 & 1 \\ 2 & 1 & 0 & 2 \\ -2 & 1 & 1 & 3 \\ 0 & 0 & 0 & 1 \end{array} \right) \begin{pmatrix} 4 \\ -2 \\ 3 \\ 1 \end{pmatrix} = \begin{pmatrix} 2 \\ 8 \\ -4 \\ 1 \end{pmatrix}.$$

Donc l'image de X est le point de coordonnées $Y = (2, 8, -4)$.

2. Si pour le même point X on avait choisi les coordonnées homogènes $X'_h = (8 : -4 : 6 : 2)$ alors on aurait obtenu

$$Y_h = A_h X'_h = \begin{pmatrix} 4 \\ 16 \\ -8 \\ 2 \end{pmatrix}$$

Mais $(4 : 16 : -8 : 2) = (2 : 8 : -4 : 1)$ et on retrouve les mêmes coordonnées $Y = (2, 8, -4) \in \mathbb{R}^3$.

3. Soit un point à l'infini de coordonnées homogènes $X_h = (v_x : v_y : v_z : 0)$. Son image

$$Y_h = A_h X_h = \begin{pmatrix} v_x - v_z \\ 2v_x + v_y \\ -2v_x + v_y + v_z \\ 0 \end{pmatrix}$$

est aussi un point à l'infini. C'est un phénomène général : un point à l'infini est envoyé sur un point à l'infini. Noter qu'en effectuant le calcul, on s'aperçoit que l'image d'un point à l'infini n'est pas affectée par la translation associée à T mais uniquement par la transformation vectorielle associée à A .

Composition

La composition de transformations affines correspond à la multiplication des matrices homogènes associées.

Proposition 7.

Si $F(X) = AX + T$ et $G(X) = BX + S$ définissent deux transformations affines et que A_h et B_h sont leurs matrices homogènes associées, alors la matrice homogène associée à $F \circ G$ (la transformation G suivie de la transformation F) est $A_h B_h$.

Proposition 8.

Si $F(X) = AX + T$ est une transformation bijective, c'est-à-dire la matrice A est inversible, alors la matrice homogène associée à F^{-1} est :

$$\left(\begin{array}{c|c} A^{-1} & -A^{-1}T \\ \hline 0 & 1 \end{array} \right).$$

Exemple.

Soit F une rotation d'axe (Oz) et d'angle $\frac{\pi}{2}$ suivie de la translation de vecteur $(1, 2, 1)$. Soit G la symétrie orthogonale par rapport au plan (Oyz) suivie d'une translation de vecteur $(1, 1, 0)$.

1. Matrices de F et G .

$$A_h = \begin{pmatrix} 0 & -1 & 0 & 1 \\ 1 & 0 & 0 & 2 \\ 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 1 \end{pmatrix} \quad B_h = \begin{pmatrix} 1 & 0 & 0 & 1 \\ 0 & -1 & 0 & 1 \\ 0 & 0 & -1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

2. Expressions de $F \circ G$ et $G \circ F$.

Par la proposition 7 ces matrices sont respectivement :

$$A_h B_h = \begin{pmatrix} 0 & 1 & 0 & 0 \\ 1 & 0 & 0 & 3 \\ 0 & 0 & -1 & 1 \\ 0 & 0 & 0 & 1 \end{pmatrix} \quad B_h A_h = \begin{pmatrix} 0 & -1 & 0 & 2 \\ -1 & 0 & 0 & -1 \\ 0 & 0 & -1 & -1 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

3. Expressions de F^{-1} et G^{-1} .

Notons \tilde{A}_h la matrice associée à F^{-1} et \tilde{B}_h la matrice associée à G^{-1} . Par la proposition 8 :

$$\tilde{A}_h = \begin{pmatrix} 0 & 1 & 0 & -2 \\ -1 & 0 & 0 & 1 \\ 0 & 0 & 1 & -1 \\ 0 & 0 & 0 & 1 \end{pmatrix} \quad \tilde{B}_h = \begin{pmatrix} 1 & 0 & 0 & -1 \\ 0 & -1 & 0 & 1 \\ 0 & 0 & -1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

Rotations de l'espace

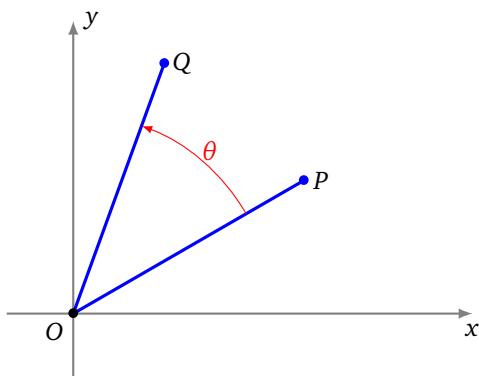
Nous étudions différentes façons d'obtenir une rotation de l'espace : en la décomposant par des rotations élémentaires ou bien à l'aide des quaternions.

1. Rotation suivant un axe

1.1. Rappels : rotation dans le plan

Commençons par exprimer une rotation dans le plan. Ce plan est muni d'un repère orthonormé direct (Oxy). Une **rotation** centrée à l'origine et d'angle θ est la transformation du plan qui envoie un point P de coordonnées (x, y) sur le point Q de coordonnées (x', y') avec :

$$\begin{cases} x' = x \cos(\theta) - y \sin(\theta) \\ y' = x \sin(\theta) + y \cos(\theta) \end{cases}$$



Il est plus facile d'exprimer une rotation à l'aide de la matrice :

$$R(\theta) = \begin{pmatrix} \cos(\theta) & -\sin(\theta) \\ \sin(\theta) & \cos(\theta) \end{pmatrix}.$$

Si on note

$$X = \begin{pmatrix} x \\ y \end{pmatrix} \quad \text{et} \quad Y = \begin{pmatrix} x' \\ y' \end{pmatrix},$$

alors l'action de la rotation s'écrit :

$$Y = RX.$$

Notations. Dans ce chapitre il va y avoir beaucoup de sinus et cosinus, nous abrégeons l'écriture par les notations suivantes :

$c_\theta = \cos(\theta)$	$s_\theta = \sin(\theta)$
---------------------------	---------------------------

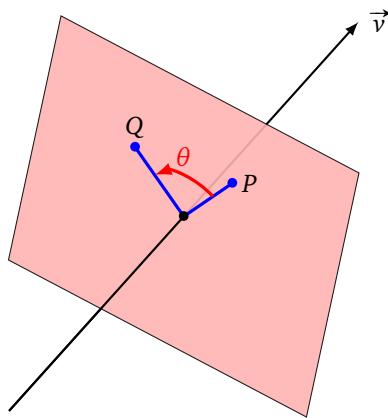
Avec ces notations la matrice de rotation R_θ s'écrit simplement :

$$R(\theta) = \begin{pmatrix} c_\theta & -s_\theta \\ s_\theta & c_\theta \end{pmatrix}.$$

Sens trigonométrique. Nos angles sont orientés : un angle positif correspond à une rotation dans le sens trigonométrique, un angle négatif à une rotation dans le sens des aiguilles d'une montre.

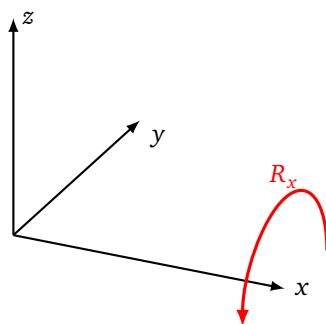
1.2. Rotation autour de l'axe x , y ou z

Une rotation de l'espace correspond à un mouvement circulaire autour d'un axe laissé fixe. Nous allons définir les rotations par leur matrice associée. Dans la suite toutes les rotations auront des axes qui passent par l'origine (on parle de rotations vectorielles). Une rotation quelconque s'obtient comme une rotation vectorielle conjuguée avec une translation.



Rotation autour de l'axe x .

$$R_x(\theta) = \begin{pmatrix} 1 & 0 & 0 \\ 0 & c_\theta & -s_\theta \\ 0 & s_\theta & c_\theta \end{pmatrix}$$



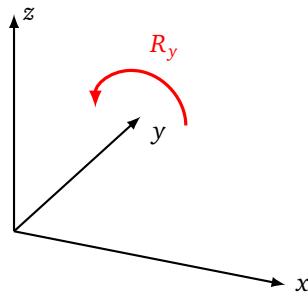
La rotation de matrice $R_x(\theta)$ envoie un point de coordonnées $X = \begin{pmatrix} x \\ y \\ z \end{pmatrix}$ sur le point de coordonnées $Y = \begin{pmatrix} x' \\ y' \\ z' \end{pmatrix}$ où :

$$Y = R_x(\theta)X.$$

Rotation autour de l'axe y .

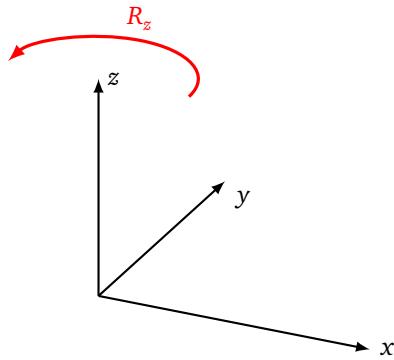
$$R_y(\theta) = \begin{pmatrix} c_\theta & 0 & s_\theta \\ 0 & 1 & 0 \\ -s_\theta & 0 & c_\theta \end{pmatrix}$$

(Attention aux signes.)

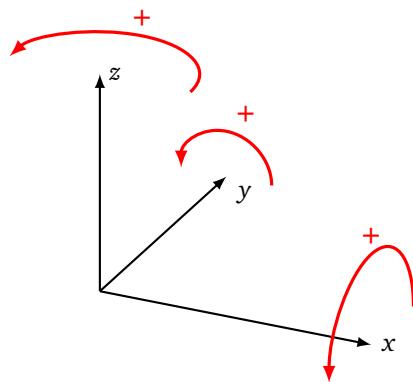


Rotation autour de l'axe z .

$$R_z(\theta) = \begin{pmatrix} c_\theta & -s_\theta & 0 \\ s_\theta & c_\theta & 0 \\ 0 & 0 & 1 \end{pmatrix}$$



Orientation. L'espace est muni d'un repère orthonormé direct ($Oxyz$). Si on se place à l'origine, à cheval sur un axe orienté dans le sens de la flèche, alors une rotation d'angle positif correspond à tourner vers la droite pour les rotations d'axe (Ox) et (Oz) et vers la gauche pour la rotation d'axe (Oy).



1.3. Rotation autour d'un axe quelconque

Une rotation vectorielle quelconque est déterminée par :

- son axe, défini par un vecteur $\vec{v} = (v_x, v_y, v_z)$,
- un angle θ .

Pour simplifier les expressions, nous supposerons que le vecteur \vec{v} est unitaire, c'est-à-dire $v_x^2 + v_y^2 + v_z^2 = 1$.

La rotation d'axe défini par \vec{v} (unitaire) et d'angle θ a pour matrice :

$$R_{\vec{v}}(\theta) = \begin{pmatrix} (1 - c_\theta)v_x^2 + c_\theta & (1 - c_\theta)v_x v_y - s_\theta v_z & (1 - c_\theta)v_x v_z + s_\theta v_y \\ (1 - c_\theta)v_x v_y + s_\theta v_z & (1 - c_\theta)v_y^2 + c_\theta & (1 - c_\theta)v_y v_z - s_\theta v_x \\ (1 - c_\theta)v_x v_z - s_\theta v_y & (1 - c_\theta)v_y v_z + v_x s_\theta & (1 - c_\theta)v_z^2 + c_\theta \end{pmatrix}$$

Ainsi cette rotation envoie un point de coordonnées $X = \begin{pmatrix} x \\ y \\ z \end{pmatrix}$ sur le point de coordonnées $Y = \begin{pmatrix} x' \\ y' \\ z' \end{pmatrix}$ par la relation $Y = R_{\vec{v}}(\theta)X$.

1.4. Formule de Rodrigues

Découvrons une autre façon d'obtenir la matrice de rotation autour d'un axe \vec{v} (unitaire) quelconque.

Notons I , $Q_{\vec{v}}$ et $P_{\vec{v}} = Q_{\vec{v}}^2$ les matrices suivantes :

$$I = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix} \quad Q_{\vec{v}} = \begin{pmatrix} 0 & -v_z & v_y \\ v_z & 0 & -v_x \\ -v_y & v_x & 0 \end{pmatrix} \quad P_{\vec{v}} = Q_{\vec{v}}^2 = \begin{pmatrix} v_x^2 - 1 & v_x v_y & v_x v_z \\ v_x v_y & v_y^2 - 1 & v_y v_z \\ v_x v_z & v_y v_z & v_z^2 - 1 \end{pmatrix}$$

$$R_{\vec{v}}(\theta) = I + \sin(\theta)Q_{\vec{v}} + (1 - \cos(\theta))Q_{\vec{v}}^2$$

Remarques :

- La matrice $Q_{\vec{v}}$ est la matrice de l'application linéaire $\vec{u} \mapsto \vec{v} \wedge \vec{u}$.
- La matrice $P_{\vec{v}}$ est la matrice de la projection orthogonale sur l'axe de rotation dirigé par \vec{v} . Elle vérifie : $P_{\vec{v}} = Q_{\vec{v}}^2 = \vec{v} \vec{v}^T$ (produit de la matrice colonne \vec{v} par la matrice ligne \vec{v}^T).
- La matrice $I - P_{\vec{v}} = I - Q_{\vec{v}}^2$ est la matrice de la projection orthogonale sur le plan orthogonal à l'axe \vec{v} .

1.5. Retrouver l'axe et l'angle

Si on nous donne une matrice R en nous affirmant que c'est une matrice de rotation, alors comment retrouver l'axe \vec{v} et l'angle θ ?

- **Retrouver un vecteur \vec{v} dirigeant l'axe.** Notons \vec{v} un vecteur propre associé à la valeur propre $\lambda = 1$, c'est-à-dire \vec{v} est un vecteur non nul solution de :

$$(R - I)\vec{v} = \vec{0}.$$

On obtient cette solution \vec{v} en résolvant un système linéaire à 3 équations et 3 inconnues. On renvoie à un cours d'algèbre linéaire pour les notions détaillées de valeurs propres/vecteurs propres.

Pour une matrice de rotation (autre que l'identité) il n'existe que deux vecteurs unitaires solutions qui sont opposés : \vec{v} et $-\vec{v}$.

- **Retrouver l'angle θ .** On obtient facilement l'angle, au signe près, par la formule :

$$\text{tr}(R) = 1 + 2 \cos(\theta)$$

où $\text{tr}(R)$ est la trace de la matrice R , c'est-à-dire la somme des éléments sur la diagonale.

Une autre méthode, consiste à choisir un vecteur \vec{u} orthogonal à l'axe \vec{v} , à calculer $R\vec{u}$, puis à calculer l'angle θ entre \vec{u} et $R\vec{u}$.

1.6. Propriétés algébriques

L'opération inverse d'une rotation d'angle θ est évidemment une rotation d'angle $-\theta$ (et de même axe) : en termes de matrices $R(\theta)^{-1} = R(-\theta)$. Toutes les matrices de rotation que nous avons vues jusqu'ici vérifient la relation $R(-\theta) = R(\theta)^T$ et donc $R(\theta)^{-1} = R(\theta)^T$. En plus nos matrices vérifient $\det(R(\theta)) = +1$, ce qui implique qu'une rotation préserve l'orientation. C'est avec ces relations qu'on définit algébriquement une matrice de rotation.

Définition.

Une matrice $R \in M_3(\mathbb{R})$ est une **matrice de rotation** si :

$$RR^T = I \quad \text{et} \quad \det(R) = +1.$$

Le **groupe spécial orthogonal** noté $SO_3(\mathbb{R})$ est l'ensemble des matrices de rotation.

Les propriétés de groupe impliquent :

Proposition 1.

Si R_1 et R_2 sont des matrices de rotations alors R_1R_2 et R_2R_1 sont aussi des matrices de rotations.

La composition de deux rotations de l'espace est une rotation de l'espace, mais ce n'est pas évident d'en connaître l'axe et l'angle. La rotation de matrice R_2R_1 correspond à l'action d'une rotation de matrice R_1 suivie d'une rotation de matrice R_2 . Alors que pour R_1R_2 c'est dans l'ordre inverse. En général R_1R_2 et R_2R_1 sont deux matrices différentes (c'est-à-dire que $SO_3(\mathbb{R})$ n'est pas un groupe commutatif).

2. Angles d'Euler

2.1. Conventions

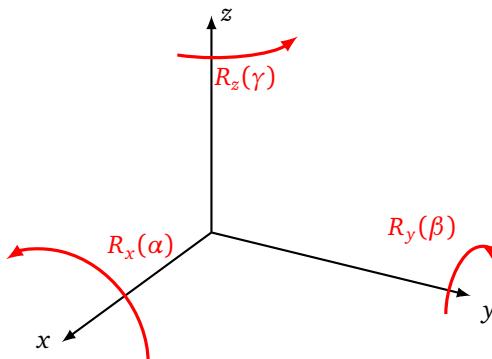
Le but est d'obtenir n'importe quelle rotation de l'espace à partir de trois rotations élémentaires. Il y a tout d'abord une difficulté technique : de nombreux choix sont possibles pour l'ordre des rotations élémentaires et il faut aussi décider si on compose les rotations d'une façon relative ou absolue. Il y a aussi une difficulté théorique (quel que soit le choix précédent) qui s'appelle le « blocage de cadran » (*gimbal lock*). Nous allons ici faire un choix (décliné en deux variantes) et donner les explications en adoptant le langage du mouvement d'un avion.

2.2. Angles d'Euler $x-y-z$

Commençons par la convention $x-y-z$ par rotations extrinsèques. On fixe un repère orthonormé direct ($Oxyz$), dit repère absolu, qui ne va pas bouger au fil des opérations. Une rotation selon la **convention $x-y-z$ par rotations extrinsèques** est une rotation qui se décompose :

$$R = R_z(\gamma)R_y(\beta)R_x(\alpha)$$

où α, β, γ sont des angles donnés.



Pour la convention $x-y-z$ on commence donc par une rotation d'axe x (d'angle α), puis d'axe y (d'angle β), et enfin d'axe z (d'angle γ). Noter bien que, pour les matrices, cet ordre correspond au produit R_x, R_y et R_z de la droite vers la gauche.

La matrice obtenue après calculs est :

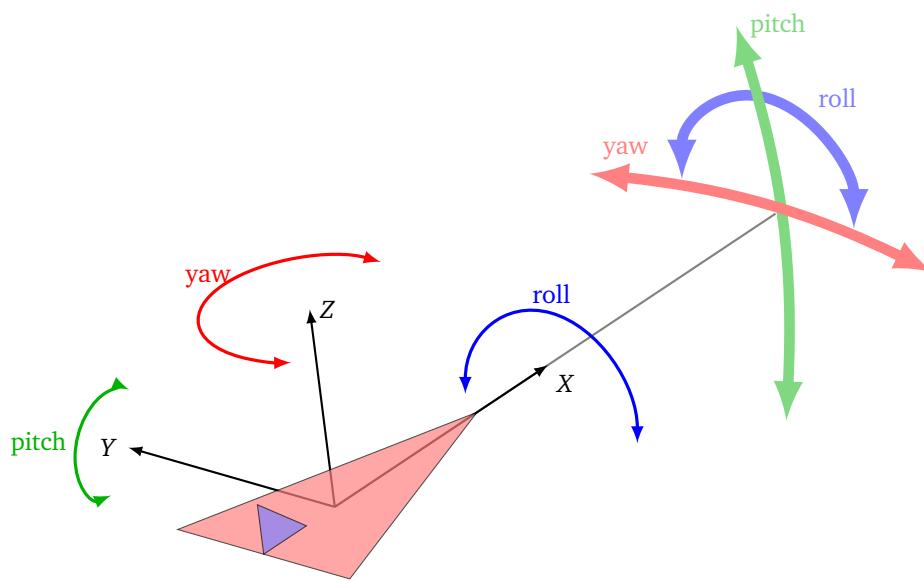
$$R = \begin{pmatrix} c_\beta c_\gamma & -c_\alpha s_\gamma + s_\alpha s_\beta c_\gamma & s_\alpha s_\gamma + c_\alpha s_\beta c_\gamma \\ c_\beta s_\gamma & c_\alpha c_\gamma + s_\alpha s_\beta s_\gamma & -s_\alpha c_\gamma + c_\alpha s_\beta s_\gamma \\ -s_\beta & s_\alpha c_\beta & c_\alpha c_\beta \end{pmatrix}$$

2.3. Mouvements d'un avion

Pour diriger un avion, le pilote dispose de trois commandes qui orientent l'avion selon trois axes. On considère un repère $(OXYZ)$, appelé repère relatif, qui est lié à l'avion (il tourne lorsque l'avion tourne). L'axe X est dirigé dans le sens de l'avion, l'axe Y le long des ailes, et l'axe des Z selon la verticale de l'avion. Les trois rotations sont :

- **Rotation d'axe Z .** Appelée ***lacet*** en aviation (*yaw*). Cela correspond au changement de direction vers la droite ou vers la gauche.
- **Rotation d'axe Y .** Appelée ***tangage*** (*pitch*). Cela correspond à monter le nez de l'avion vers le haut ou bien à le descendre vers le bas.
- **Rotation d'axe X .** Appelée ***roulis*** (*roll*). Cela correspond à monter l'aile droite et baisser l'aile gauche, ou l'inverse, alors que l'axe longitudinal de l'avion reste fixe.

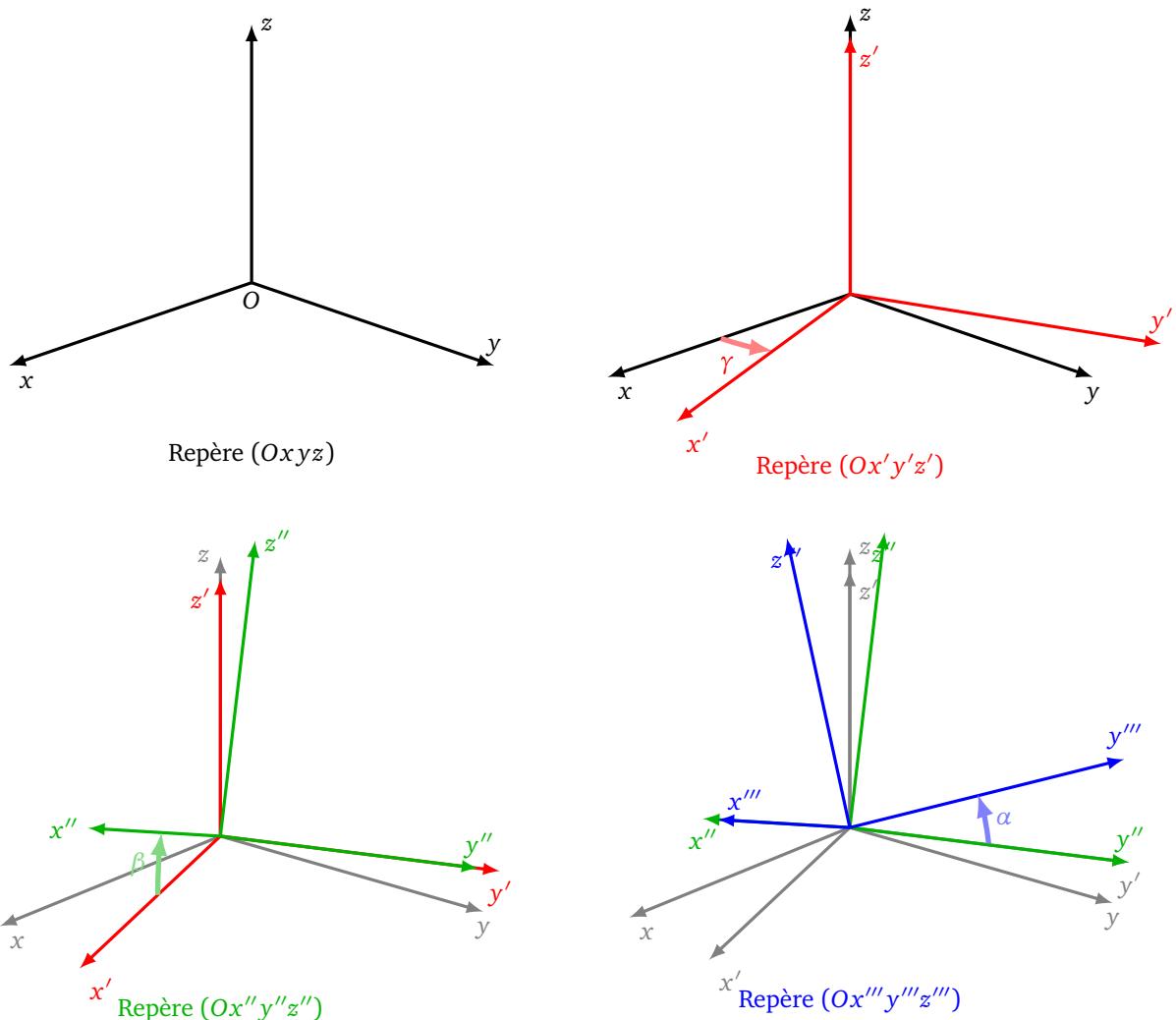
Rappelons que le repère $(OXYZ)$ est lié à l'avion, après chaque rotation le repère change de position par rapport à un observateur situé au sol.



2.4. Angles d'Euler $z-y'-x''$

Expliquons la ***convention $z-y'-x''$ par rotations intrinsèques***. On fixe un repère orthonormé direct $(Oxyz)$, dit repère absolu, qui ne va pas bouger au fil des opérations.

1. On commence par une rotation \mathcal{R}_1 autour de l'axe z et d'angle γ . On considère maintenant le repère $(Ox'y'z')$ obtenu par rotation du repère $(Oxyz)$.
2. On continue par une rotation \mathcal{R}_2 autour de l'axe y' et d'angle β . On considère ensuite le repère $(Ox''y''z'')$ obtenu par rotation du repère $(Ox'y'z')$.
3. On termine par la rotation \mathcal{R}_3 autour de l'axe x'' et d'angle α .



C'est très facile de comprendre si on se place dans le repère $(OXYZ)$ relatif, lié à l'objet en mouvement. Reprenons le cas d'un avion, alors on effectue successivement une rotation autour de l'axe relatif Z (*yaw*), puis de l'axe relatif Y (*pitch*), et enfin de l'axe relatif X (*roll*). On pourrait exprimer cette rotation par la notation matricielle $R_z(\gamma)R_y(\beta)R_x(\alpha)$, mais cela peut être trompeur car à chaque rotation le repère $(OXYZ)$ en jeu a changé.

2.5. Équivalence entre $x-y-z$ et $z-y'-x''$

Nous allons montrer que les conventions $x-y-z$ et $z-y'-x''$ sont équivalentes :

$$R_z(\gamma)R_y(\beta)R_x(\alpha) = R_{x''}(\alpha)R_{y'}(\beta)R_z(\gamma)$$

Les calculs sont plutôt théoriques et peuvent être passés lors d'une première lecture. On commence par des rappels sur les changements de base.

Passage d'une base à une autre pour les coordonnées d'un point/vecteur. On considère la base canonique \mathcal{B} de \mathbb{R}^3 formée de trois vecteurs $(\vec{i}, \vec{j}, \vec{k})$. Autrement dit, on se place dans le repère $(Oxyz)$ usuel, considéré comme repère absolu. On considère une autre base \mathcal{B}' formée de trois vecteurs $(\vec{f}_1, \vec{f}_2, \vec{f}_3)$, cela correspond à un nouveau repère $(Ox'y'z')$.

Notons P , la matrice de passage de l'ancienne base \mathcal{B} à la nouvelle base \mathcal{B}' . Comme ici \mathcal{B} est la base canonique, le premier vecteur colonne de P est formé des coordonnées du vecteur \vec{f}_1 , le deuxième vecteur colonne est formé des coordonnées de \vec{f}_2 , le troisième vecteur colonne est formé des coordonnées de \vec{f}_3 .

Notons X les coordonnées d'un point (ou d'un vecteur) dans la base \mathcal{B} et notons X' les coordonnées de ce même point dans la base \mathcal{B}' , X et X' sont reliés par la relation suivante :

$$X = PX'.$$

Passage d'une base à une autre pour une matrice/application linéaire. Soit $f : \mathbb{R}^3 \rightarrow \mathbb{R}^3$ une application linéaire (par exemple une rotation). On note A la matrice de f dans la base \mathcal{B} et on note B la matrice de f dans la base \mathcal{B}' . Ces deux matrices sont liées par la relation suivante :

$$A = PBP^{-1}.$$

Passage de la convention $z\text{-}y'\text{-}x''$ à la convention $x\text{-}y\text{-}z$.

Exprimons les rotations de la convention $z\text{-}y'\text{-}x''$ dans le repère fixe $(Oxyz)$.

- On se place dans la base \mathcal{B} (c'est-à-dire avec le repère $(Oxyz)$). On applique une rotation d'axe z . Cette application a pour matrice $A_1 = R_z(\gamma)$ dans la base \mathcal{B} .
- Cette rotation transforme le repère $(Oxyz)$ en un repère $(Ox'y'z')$ (associé à la base \mathcal{B}'). La matrice de passage correspondante P_2 est tout simplement $P_2 = R_z(\gamma)$.
- On se place dans le repère $(Ox'y'z')$ et on applique une rotation d'axe y' . Dans ce repère, la matrice de cette rotation est $B_2 = R_y(\beta)$. Quelle est la matrice de cette même rotation dans le repère fixe $(Oxyz)$? C'est $A_2 = P_2 B_2 P_2^{-1}$ d'après la formule de changement de base. Ainsi $A_2 = R_z R_y R_z^{-1}$ (en omettant les angles).
- Quelle est la matrice de la rotation autour de z suivie d'une rotation autour de y' ? Dans le repère fixe $(Oxyz)$ cette matrice est $A_2 A_1$. Calculons-la (en omettant β et γ) :

$$A_2 A_1 = (R_z R_y R_z^{-1}) R_z = R_z R_y.$$

Ainsi la convention tronquée $z\text{-}y'$ correspond à la convention $y\text{-}z$.

- La rotation autour de l'axe y' transforme le repère $(Ox'y'z')$ en un repère $(Ox''y''z'')$. La matrice de passage associée est $P_3 = R_y(\beta)$.
- Dans le repère $(Ox''y''z'')$ on applique la rotation $B_3 = R_x(\alpha)$. Dans le repère $(Ox'y'z')$, la matrice de cette même rotation est $A'_3 = P_3 B_3 P_3^{-1}$. Et dans le repère fixe $(Oxyz)$, c'est

$$A_3 = P_2 A'_3 P_2^{-1} = P_2 (P_3 B_3 P_3^{-1}) P_2^{-1} = R_z R_y R_x R_y^{-1} R_z^{-1}.$$

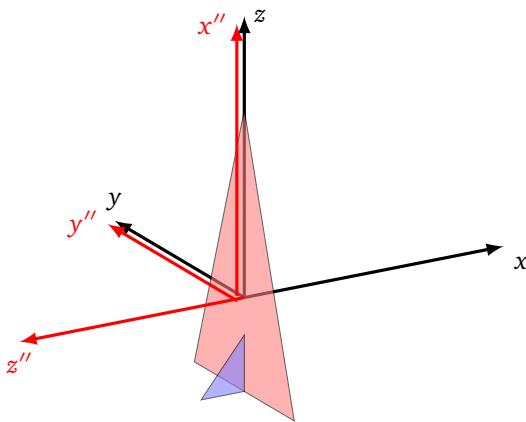
- Ainsi la composition des trois rotations avec la convention $z\text{-}y'\text{-}x''$ dans le repère fixe $(Oxyz)$ a pour matrice $A_3 A_2 A_1$, dont le calcul se simplifie :

$$A_3 A_2 A_1 = (R_z R_y R_x R_y^{-1} R_z^{-1})(R_z R_y R_z^{-1}) R_z = R_z(\gamma) R_y(\beta) R_x(\alpha).$$

Ce qui est exactement la matrice de la convention $x\text{-}y\text{-}z$!

2.6. Blocage de cadran

Expliquons le **blocage de cadran** (*gimbal lock*) avec la convention $z\text{-}y'\text{-}x''$. Imaginons un avion en position verticale (« chandelle ») dans un repère absolu $(Oxyz)$. Pour le repère $(Ox''y''z'')$ lié à l'avion, l'axe x'' coïncide avec l'axe z . Il y a normalement trois rotations possibles avec la convention $z\text{-}y'\text{-}x''$, mais ici comme les axes z et x'' sont confondus, on a perdu un degré de liberté, il n'y a en fait que deux rotations possibles (une autour de l'axe y' et une autour de l'axe z qui est aussi x''). Bien sûr il suffit d'effectuer une rotation d'axe y' pour séparer de nouveau les axes z et x'' .



Lors de la mission Apollo 11 qui envoya Neil Armstrong et Buzz Aldrin sur la Lune, le gyroscope de Mike Collins, resté en orbite, resta bloqué à cause du phénomène décrit ci-dessus. Ce gyroscope était naturellement conçu à l'aide de trois cadrants circulaires (un pour chaque axe de rotation). Une solution pour éviter ce problème est de rajouter un quatrième degré de liberté qui permet de ne pas s'approcher des points de blocage. C'est pourquoi après sa mission Mike Collins déclara « Pour Noël prochain j'aimerais un quatrième cadran ! ».

2.7. Retrouver les angles d'Euler

On nous donne une matrice de rotation R . Comment décomposer cette matrice selon la convention x - y - z :

$$R = R_z(\gamma)R_y(\beta)R_x(\alpha) ?$$

Notons r_{ij} les coefficients de R :

$$R = \begin{pmatrix} r_{11} & r_{12} & r_{13} \\ r_{21} & r_{22} & r_{23} \\ r_{31} & r_{32} & r_{33} \end{pmatrix}$$

Alors, en identifiant cette matrice avec la matrice obtenue au paragraphe 2.2, on obtient d'abord :

$$\beta = -\arcsin(r_{31})$$

(car $r_{31} = -s_\beta$). Ensuite, on discute selon la valeur de β :

- Si $\beta \in]-\frac{\pi}{2}, +\frac{\pi}{2}[$ alors

$$\alpha = \text{arctan2}(r_{32}, r_{33}) \quad \text{et} \quad \gamma = \text{arctan2}(r_{21}, r_{11}).$$

- Si $\beta = +\frac{\pi}{2}$ alors $\alpha - \gamma = \text{arctan2}(-r_{23}, r_{22})$. On dispose d'un degré de liberté dans le choix des angles, la décomposition n'est donc pas unique.
- Si $\beta = -\frac{\pi}{2}$ alors $\alpha + \gamma = \text{arctan2}(-r_{23}, r_{22})$. On dispose encore d'un degré de liberté, la décomposition n'est pas unique.

Pour les deux cas particuliers $\beta = \pm\frac{\pi}{2}$, on retrouve le phénomène de blocage de cadran, qui se traduit ici par l'absence d'une décomposition unique pour certaines configurations.

Exercice.

On veut étudier en détail et à la main le phénomène de blocage de cadran. Dans cet exercice on fixe :

$$\beta = +\frac{\pi}{2}.$$

Soient α, γ des angles quelconques et soit R la matrice de rotation de la convention x - y - z :

$$R = R_z(\gamma)R_y(\beta)R_x(\alpha)$$

1. Vérifier que :

$$R = \begin{pmatrix} 0 & s_{\alpha-\gamma} & c_{\alpha-\gamma} \\ 0 & c_{\alpha-\gamma} & -s_{\alpha-\gamma} \\ -1 & 0 & 0 \end{pmatrix}$$

où $c_{\alpha-\gamma} = \cos(\alpha - \gamma)$ et $s_{\alpha-\gamma} = \sin(\alpha - \gamma)$.

2. Résoudre l'équation $RX = X$ où $X = \begin{pmatrix} x \\ y \\ z \end{pmatrix}$ et en déduire que l'axe de la rotation est dirigé par le vecteur (non unitaire) :

$$\vec{v} = \begin{pmatrix} 1 - c_{\alpha-\gamma} \\ s_{\alpha-\gamma} \\ c_{\alpha-\gamma} - 1 \end{pmatrix}.$$

3. À l'aide de la trace de matrices, montrer que l'angle θ de la rotation R vérifie :

$$|\theta| = \arccos\left(\frac{\cos(\alpha - \gamma) - 1}{2}\right).$$

4. Expliquer pourquoi la décomposition de R n'est pas unique dans le cas $\beta = \frac{\pi}{2}$.

3. Quaternions

3.1. Motivation

Les rotations définies par un axe et un angle ou bien définies à l'aide des angles d'Euler sont assez difficiles à manipuler en particulier si on souhaite composer deux rotations. En plus avec les angles d'Euler, on risque de se confronter au « blocage de cadran », inévitable pour certaines configurations. La solution élégante à tous ces problèmes est d'utiliser les quaternions. Les calculs sont de simples manipulations algébriques, faciles à réaliser pour un humain et un ordinateur. L'inconvénient est que l'on perd en compréhension géométrique et que la notion n'est presque jamais enseignée. On va présenter ici cette nouvelle notion de quaternions, qui sera vite comprise par tous ceux qui connaissent les nombres complexes.

3.2. Rappels : nombres complexes

Les quaternions sont analogues aux nombres complexes, en un petit peu plus compliqués. Rappelons qu'un nombre complexe est l'écriture :

$$z = a + bi$$

où $a, b \in \mathbb{R}$ et i vérifient :

$$i^2 = -1.$$

Très rapidement :

- On identifie un nombre complexe $z = a + bi$ à un point (a, b) du plan.
- a s'appelle la partie réelle et b la partie imaginaire.
- Le module de z est le réel positif défini par $|z|^2 = a^2 + b^2$.
- Le conjugué est $\bar{z} = a - bi$, de sorte que $|z|^2 = z\bar{z}$.
- La multiplication est commutative : $z_1 z_2 = z_2 z_1$.
- On note $e^{i\theta} = \cos(\theta) + i \sin(\theta)$.
- L'opération $z \mapsto ze^{i\theta}$, correspond à transformer le point (a, b) par la rotation d'angle θ , centrée à l'origine.

3.3. Écriture algébrique des quaternions

Un quaternion est l'écriture :

$$q = a + bi + cj + dk$$

où a, b, c, d sont des réels et où i, j, k vérifient :

$$i^2 = j^2 = k^2 = ijk = -1$$

Avant d'expliquer comment multiplier deux quaternions, il est fondamental de comprendre que la multiplication des quaternions n'est pas commutative. Par exemple $ij = k$ alors que $ji = -k$. Voici un tableau qui résume les multiplications élémentaires (colonne de gauche premier terme, première ligne second terme, dans le tableau le produit des deux) :

\times	i	j	k
i	-1	k	-j
j	-k	-1	i
k	j	-i	-1

Ce tableau se déduit des axiomes. Par exemple, exprimons ij . On sait $ijk = -1$, donc en multipliant à droite les deux termes de cette égalité par k on obtient $ijk^2 = -k$ mais comme $k^2 = -1$ on obtient $ij = k$. À vous de vérifier les autres résultats.

Une fois les multiplications élémentaires comprises, la multiplication de deux quaternions s'effectue comme un produit où i, j et k jouent le rôle de variables.

Exemple.

Soient :

$$q_1 = 1 + 2i - 3j \quad \text{et} \quad q_2 = i - 4k.$$

Alors :

$$\begin{aligned} q_1 q_2 &= (1 + 2i - 3j)(i - 4k) \\ &= i - 4k + 2i^2 - 8ik - 3ji + 12jk \\ &= i - 4k - 2 + 8j + 3k + 12i \\ &= -2 + 13i + 8j - k \end{aligned}$$

Par contre :

$$\begin{aligned} q_2 q_1 &= (i - 4k)(1 + 2i - 3j) \\ &= i + 2i^2 - 3ij - 4k - 8ki + 12kj \\ &= i - 2 - 3k - 4k - 8j - 12i \\ &= -2 - 11i - 8j - 7k \end{aligned}$$

Donc $q_1 q_2$ n'est pas égal à $q_2 q_1$.

3.4. Propriétés

On retient de l'exemple précédent :

La multiplication des quaternions n'est pas commutative.

Voici quelques notions et propriétés élémentaires. On note toujours $q = a + bi + cj + dk$.

- a s'appelle la **partie réelle** et $bi + cj + dk$ s'appelle la **partie vectorielle** (ou aussi **partie imaginaire**).
- La **norme** $\|q\|$ est le réel positif défini par :

$$\|q\|^2 = a^2 + b^2 + c^2 + d^2.$$

- Le **conjugué** de q est $\bar{q} = a - bi - cj - dk$. De sorte que $q\bar{q} = \|q\|^2$.
- Pour un quaternion non nul (c'est-à-dire $(a, b, c, d) \neq (0, 0, 0, 0)$),

$$q^{-1} = \frac{1}{\|q\|^2} \bar{q} = \frac{a - bi - cj - dk}{a^2 + b^2 + c^2 + d^2}$$

de sorte que $qq^{-1} = q^{-1}q = 1$.

- Si la partie réelle d'un quaternion est nulle (c'est-à-dire $a = 0$), on parle de **quaternion vectoriel pur**. Le produit de deux quaternions vectoriels purs est un quaternion vectoriel pur.

3.5. Rotations

Soit un point P de coordonnées $(x, y, z) \in \mathbb{R}^3$, que l'on peut aussi considérer comme un vecteur. À P on associe le quaternion vectoriel pur :

$$p = xi + yj + zk.$$

Réiproquement tout quaternion vectoriel pur $xi + yj + zk$ est identifié au point de coordonnées (x, y, z) ou aussi au vecteur $\begin{pmatrix} x \\ y \\ z \end{pmatrix}$.

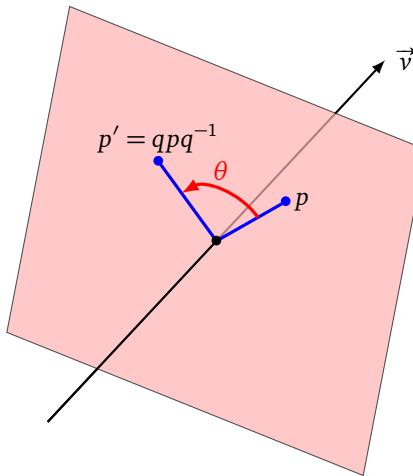
Considérons une rotation \mathcal{R} d'axe le vecteur unitaire $\vec{v} = (v_x, v_y, v_z)$ et d'angle θ . À cette rotation on associe le quaternion :

$$q = \cos\left(\frac{\theta}{2}\right) + \sin\left(\frac{\theta}{2}\right)(v_x i + v_y j + v_z k)$$

Notons P' l'image de P par la rotation \mathcal{R} et p' le quaternion vectoriel pur associé. Alors

$$p' = qpq^{-1}$$

On dit que p' est le **conjugué** de p par q . Ainsi le calcul de l'image d'un point par une rotation correspond à une simple multiplication de quaternions. Noter que comme q est unitaire alors $q^{-1} = \bar{q}$.



La preuve que ce calcul correspond à une rotation n'est pas si simple (et nous l'admettons). Par contre cela entraîne une formule simple pour la composition de deux rotations.

Proposition 2.

Si une rotation \mathcal{R}_1 a pour quaternion q_1 et une rotation \mathcal{R}_2 a pour quaternion q_2 , alors $\mathcal{R}_2 \circ \mathcal{R}_1$ (la rotation \mathcal{R}_1 suivie de la rotation \mathcal{R}_2) a pour quaternion associé $q_2 q_1$. Cela correspond à la transformation :

$$p \mapsto (q_2 q_1) p (q_2 q_1)^{-1}.$$

Démonstration. La rotation \mathcal{R}_1 est la transformation $p \mapsto p' = q_1 p q_1^{-1}$. La rotation \mathcal{R}_2 est la transformation $p' \mapsto p'' = q_2 p' q_2^{-1}$. La rotation \mathcal{R}_1 suivie de la rotation \mathcal{R}_2 est donc $p \mapsto p'' = q_2(q_1 p q_1^{-1})q_2^{-1} = (q_2 q_1)p(q_2 q_1)^{-1}$. \square

Exemple.

Soit \mathcal{R}_1 une rotation d'axe $(-2, 1, 1)$ et d'angle $\frac{\pi}{3}$. Soit \mathcal{R}_2 une rotation d'axe $(1, 0, -1)$ et d'angle $\frac{\pi}{2}$. Soit P le point de coordonnées $(1, 0, 0)$. Déterminer l'image de P par la rotation \mathcal{R}_1 suivie de la rotation \mathcal{R}_2 .

- On commence par rendre unitaire le vecteur de l'axe de \mathcal{R}_1 : $\vec{v}_1 = \frac{\sqrt{6}}{6}(-2, 1, 1)$. Le quaternion associé à \mathcal{R}_1 est

$$q_1 = \cos\left(\frac{\pi}{6}\right) + \sin\left(\frac{\pi}{6}\right)\frac{\sqrt{6}}{6}(-2i + j + k) = \frac{\sqrt{3}}{2} - \frac{\sqrt{6}}{6}i + \frac{\sqrt{6}}{12}j + \frac{\sqrt{6}}{12}k.$$

- On normalise le vecteur de l'axe de \mathcal{R}_2 : $\vec{v}_2 = \frac{\sqrt{2}}{2}(1, 0, -1)$. Le quaternion associé à \mathcal{R}_2 est

$$q_2 = \cos\left(\frac{\pi}{4}\right) + \sin\left(\frac{\pi}{4}\right)\frac{\sqrt{2}}{2}(i - k) = \frac{\sqrt{2}}{2} + \frac{1}{2}i - \frac{1}{2}k.$$

- Le quaternion associé au point P est simplement $p = i$.

Passons aux calculs.

1. Notons $P' = \mathcal{R}_1(P)$. Son quaternion associé est (après calculs) :

$$p' = q_1 p q_1^{-1} = \frac{5}{6}i + \left(-\frac{1}{6} + \frac{1}{4}\sqrt{2}\right)j + \left(-\frac{1}{6} - \frac{1}{4}\sqrt{2}\right)k.$$

2. Notons $P'' = \mathcal{R}_2(P') = \mathcal{R}_2 \circ \mathcal{R}_1(P)$, le quaternion associé (après calculs) est :

$$p'' = q_2 p' q_2^{-1} = \frac{1}{2}i - \frac{\sqrt{2}}{2}j - \frac{1}{2}k.$$

Ainsi les coordonnées de P'' sont $(\frac{1}{2}, -\frac{\sqrt{2}}{2}, -\frac{1}{2})$.

On aurait obtenu le même résultat si on avait d'abord calculé $q_2 q_1$, puis $(q_2 q_1)^{-1}$, et enfin $p'' = (q_2 q_1)p(q_2 q_1)^{-1}$ (voir l'exemple ci-après).

De même on prouve facilement les résultats suivants.

Proposition 3.

Soit \mathcal{R} la rotation (d'angle θ) associée à un quaternion unitaire q .

1. La rotation inverse (d'angle $-\theta$ et de même axe) est associée à q^{-1} (qui est aussi \bar{q} car q est unitaire), c'est-à-dire à la transformation $p \mapsto q^{-1}pq$.
2. La rotation \mathcal{R} itérée n fois (donc une rotation d'angle $n\theta$) est associée à q^n , c'est-à-dire à la transformation $p \mapsto q^n p q^{-n}$.

3.6. Retrouver l'axe et l'angle

Si on nous donne un quaternion unitaire $q = a + bi + cj + dk$, alors on retrouve l'axe \vec{v} et l'angle θ selon les formules suivantes :

$$\vec{v} = \frac{1}{\sqrt{b^2 + c^2 + d^2}} \begin{pmatrix} b \\ c \\ d \end{pmatrix} \quad \theta = 2 \arctan2\left(\sqrt{b^2 + c^2 + d^2}, a\right).$$

Exemple.

Soit la rotation \mathcal{R}_1 d'axe $(1, 1, 0)$ et d'angle π et la rotation \mathcal{R}_2 d'axe $(0, 1, 1)$ et d'angle π (ce sont deux retournements). Quels sont l'axe et l'angle de la rotation $\mathcal{R}_2 \circ \mathcal{R}_1$?

1. On commence par rendre les vecteurs unitaires : $\vec{v}_1 = \frac{\sqrt{2}}{2}(1, 1, 0)$ et $\vec{v}_2 = \frac{\sqrt{2}}{2}(0, 1, 1)$. Les quaternions q_1 et q_2 associés respectivement à \mathcal{R}_1 et \mathcal{R}_2 sont alors :

$$q_1 = \frac{\sqrt{2}}{2}i + \frac{\sqrt{2}}{2}j \quad q_2 = \frac{\sqrt{2}}{2}j + \frac{\sqrt{2}}{2}k.$$

2. Le quaternion associé à la rotation $\mathcal{R}_2 \circ \mathcal{R}_1$ est :

$$q = q_2 q_1 = -\frac{1}{2} - \frac{1}{2}\mathbf{i} + \frac{1}{2}\mathbf{j} - \frac{1}{2}\mathbf{k}.$$

3. On applique les formules énoncées précédemment pour retrouver l'axe et l'angle :

$$\vec{v} = \frac{2}{\sqrt{3}} \begin{pmatrix} -\frac{1}{2} \\ \frac{1}{2} \\ -\frac{1}{2} \end{pmatrix} = \frac{\sqrt{3}}{3} \begin{pmatrix} -1 \\ 1 \\ -1 \end{pmatrix}$$

et

$$\theta = 2 \arctan 2 \left(\frac{\sqrt{3}}{2}, -\frac{1}{2} \right) = \frac{4\pi}{3}.$$

Ainsi $\mathcal{R}_2 \circ \mathcal{R}_1$ est une rotation d'axe $(-1, 1, -1)$ et d'angle $\frac{4\pi}{3} = -\frac{2\pi}{3}[2\pi]$, ce qui était loin d'être évident !

Perspective

Nous expliquons différentes façons de représenter l'espace 3D sur un plan 2D.

1. Perspective linéaire

La perspective linéaire est aussi appelée *perspective conique* ou *projection centrale*.

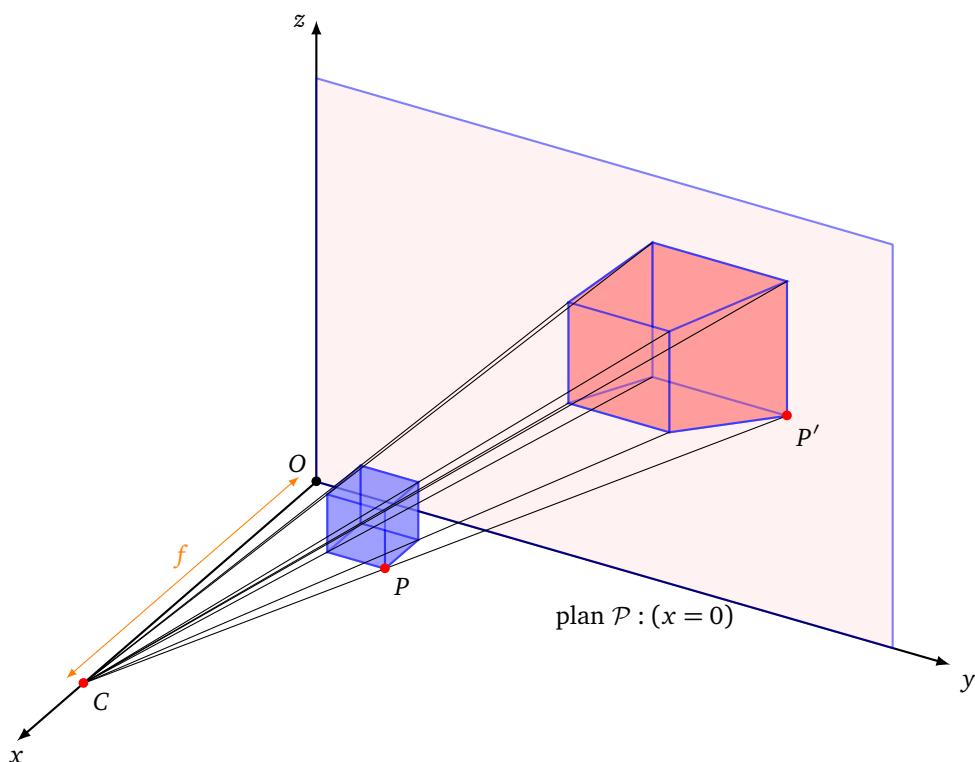
1.1. Principe

La perspective linéaire est la perspective qui se rapproche le plus de la vision humaine, les objets éloignés paraissent plus petits.

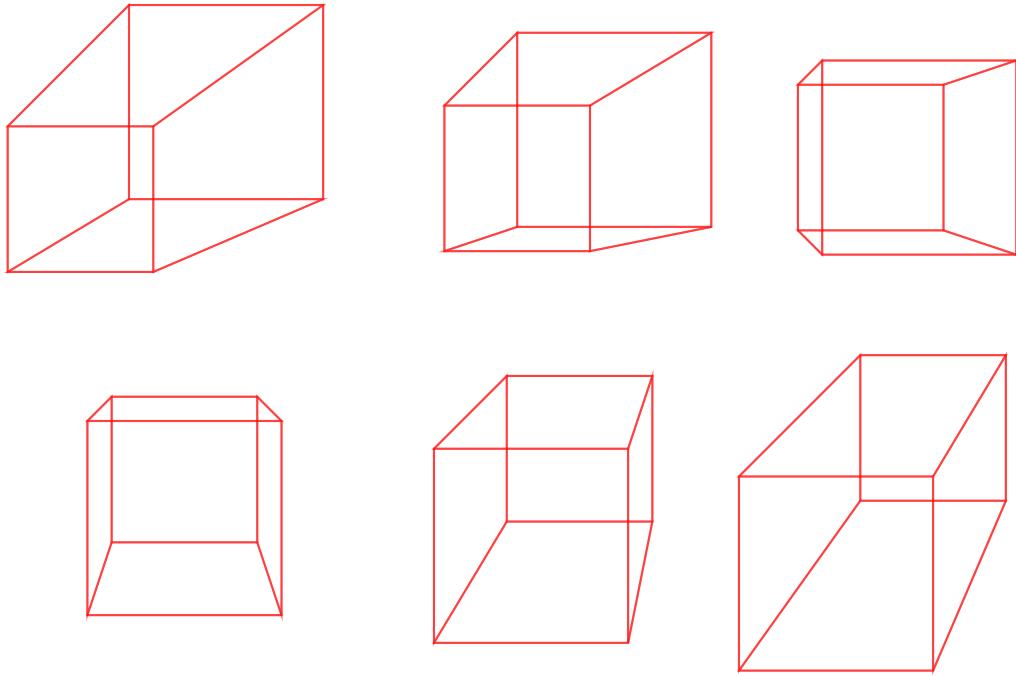
Notations :

- C est un point de l'espace, il peut représenter la caméra, l'œil, un capteur, ou aussi un éclairage,
- \mathcal{P} est un plan de l'espace,
- P désigne un point d'un objet \mathcal{O} .

Construction : pour chaque point P , si la droite (CP) recoupe le plan \mathcal{P} , on note P' ce point d'intersection. On définit ainsi une projection de l'espace \mathcal{E} sur le plan \mathcal{P} .

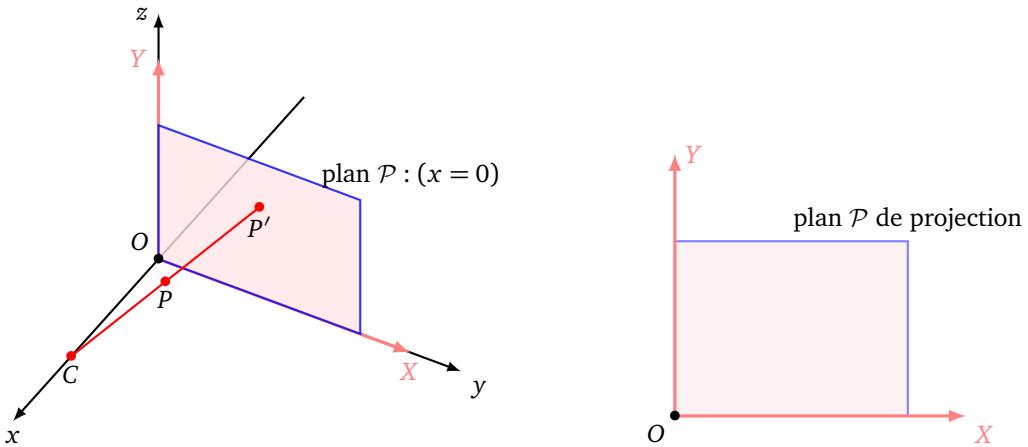


Voici quelques projections du même cube mais avec différents points C .



1.2. Cas simple

L'espace est muni du repère orthonormé $(Oxyz)$ classique. Le point $C(f, 0, 0)$ est fixé sur l'axe (Ox) (la valeur f s'appelle la *focale*). Le plan \mathcal{P} de projection est le plan (Oyz) , c'est-à-dire d'équation $(x = 0)$. Dans ce plan on note (X, Y) les coordonnées du plan (avec $X = y$ et $Y = z$).



Proposition 1.

Pour P de coordonnées (x, y, z) (avec $x \neq f$) alors son image par la perspective linéaire est P' de coordonnées (X, Y) dans \mathcal{P} avec :

$$\boxed{\begin{cases} X = \lambda y \\ Y = \lambda z \end{cases} \quad \text{où} \quad \lambda = \frac{-f}{x-f}}$$

Démonstration. Les points C, P et P' sont alignés donc il existe $\lambda \in \mathbb{R}$ tel que $\overrightarrow{CP'} = \lambda \overrightarrow{CP}$. Notons $P'(0, y', z')$ les coordonnées de P' dans le repère $(Oxyz)$. Alors :

$$\overrightarrow{CP} = \begin{pmatrix} x-f \\ y \\ z \end{pmatrix} \quad \overrightarrow{CP'} = \begin{pmatrix} -f \\ y' \\ z' \end{pmatrix}$$

Ainsi

$$\overrightarrow{CP'} = \lambda \overrightarrow{CP} \iff \begin{cases} -f &= \lambda(x-f) \\ y' &= \lambda y \\ z' &= \lambda z \end{cases} \iff \begin{cases} \lambda &= \frac{-f}{x-f} \\ X &= \lambda y \\ Y &= \lambda z \end{cases}$$

où l'on a identifié $X = y'$ et $Y = z'$. □

1.3. Cas général

Voici les formules, pour le cas d'un point $C(x_C, y_C, z_C)$ de coordonnées quelconques, en projetant toujours sur le plan (Oyz),

Proposition 2.

Pour P de coordonnées (x, y, z) (avec $x \neq x_C$) alors son image par la perspective linéaire est P' de coordonnées (X, Y) dans \mathcal{P} avec :

$$\begin{cases} X &= \lambda(y - y_C) + y_C \\ Y &= \lambda(z - z_C) + z_C \end{cases} \quad \text{où} \quad \lambda = \frac{-x_C}{x - x_C}$$

Démonstration. Depuis l'identité $\overrightarrow{CP'} = \lambda \overrightarrow{CP}$, on a :

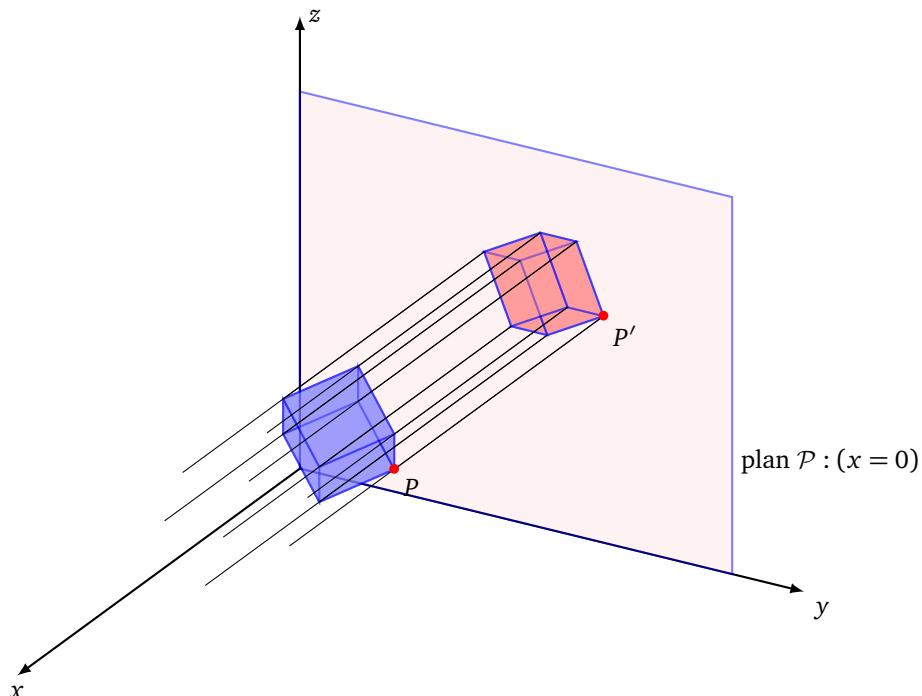
$$\begin{pmatrix} 0 - x_C \\ y' - y_C \\ z' - z_C \end{pmatrix} = \lambda \begin{pmatrix} x - x_C \\ y - y_C \\ z - z_C \end{pmatrix},$$

d'où les formules. □

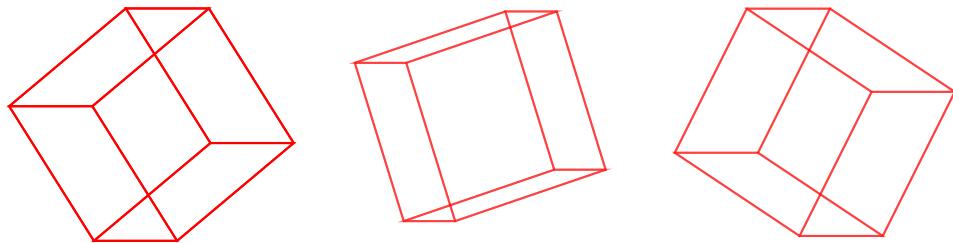
2. Projection orthogonale

2.1. Principe

C'est comme si on plaçait une caméra à une distance infinie, dirigée orthogonalement à un plan. Une fois projetés, deux objets identiques auront les mêmes dimensions, qu'ils soient proches ou éloignés du plan de projection. La projection orthogonale est utilisée dans le dessin technique, les plans d'architectes...

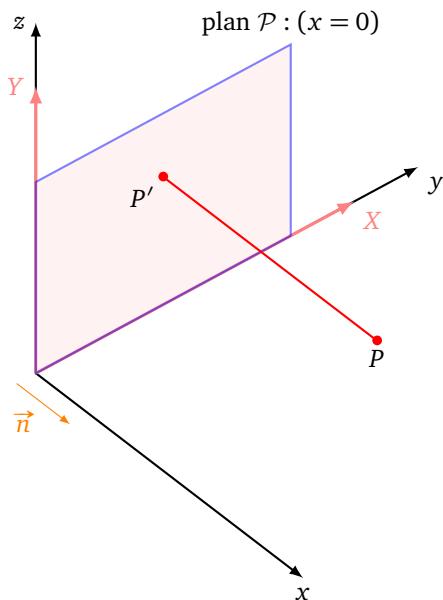


Voici quelques projections obtenues en faisant tourner un cube.



2.2. Cas simple

Le plan de projection \mathcal{P} est le plan (Oyz) et l'axe de projection est l'axe (Ox) . Le vecteur $\vec{n} = \begin{pmatrix} 1 \\ 0 \\ 0 \end{pmatrix}$ est colinéaire à cet axe et orthogonal au plan.



Si on considère la projection comme une application de $\mathbb{R}^3 \rightarrow \mathbb{R}^3$ alors elle s'écrit :

$$P = \begin{pmatrix} x \\ y \\ z \end{pmatrix} \longmapsto P' = \begin{pmatrix} 0 \\ y \\ z \end{pmatrix}.$$

C'est une application linéaire, dont la matrice est :

$$M_1 = \begin{pmatrix} 0 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix}.$$

Si on considère cette projection de l'espace vers le plan \mathcal{P} dont les coordonnées sont (X, Y) , alors les formules sont tout simplement :

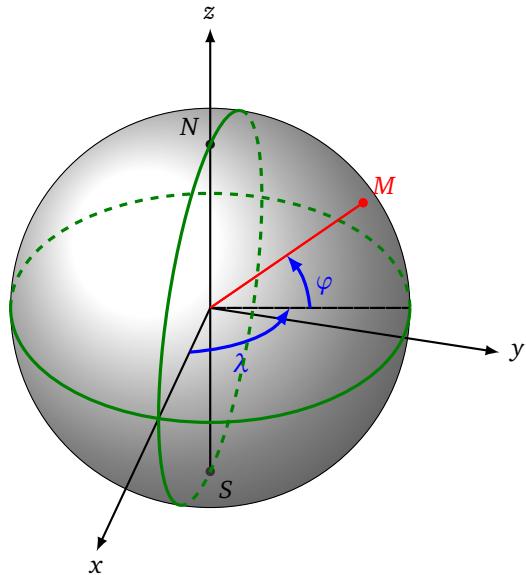
$$X = y \quad \text{et} \quad Y = z.$$

La matrice correspondante est :

$$M'_1 = \begin{pmatrix} 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix}.$$

2.3. Cas général

On considère une projection orthogonale sur un plan quelconque (passant par l'origine). Ce plan est déterminé par un vecteur normal \vec{n} , qui est aussi la direction de projection. Nous allons caractériser cette direction par un vecteur déterminé par ses coordonnées longitude/latitude. Ainsi une projection orthogonale est déterminée par le vecteur $\vec{n} = \overrightarrow{OM}$ où le point $M(1, \varphi, \lambda)$ est défini en coordonnées sphériques (par sa latitude φ et sa longitude λ).



On note (X, Y) les coordonnées sur le plan de projection.

Proposition 3.

La projection orthogonale suivant l'axe de latitude φ et de longitude λ est donnée par l'application : $\begin{pmatrix} x \\ y \\ z \end{pmatrix} \mapsto \begin{pmatrix} X \\ Y \\ z \end{pmatrix}$ définie par :

$$\begin{pmatrix} X \\ Y \\ z \end{pmatrix} = M \begin{pmatrix} x \\ y \\ z \end{pmatrix} \quad \text{où} \quad M = \begin{pmatrix} -\sin \lambda & \cos \lambda & 0 \\ \sin \varphi \cos \lambda & \sin \varphi \sin \lambda & \cos \varphi \end{pmatrix}.$$

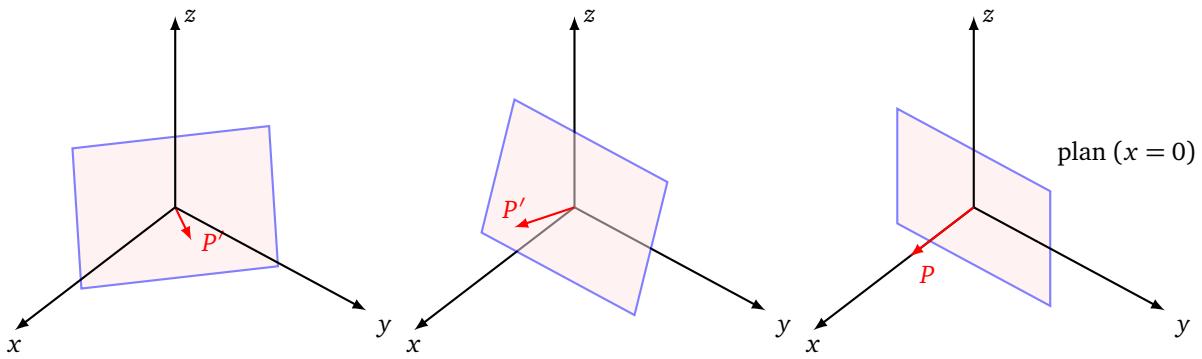
Autrement dit :

$$\begin{cases} X &= -\sin(\lambda) \cdot x + \cos(\lambda) \cdot y \\ Y &= \sin(\varphi) \cos(\lambda) \cdot x + \sin(\varphi) \sin(\lambda) \cdot y + \cos(\varphi) \cdot z \end{cases}$$

Démonstration. Il s'agit de se ramener au cas simple, c'est-à-dire à la projection orthogonale sur le plan (Oyz) dont la matrice est M'_1 .

- Tout d'abord on modifie le plan de projection, en changeant la longitude de sorte que la trace du plan de projection sur le plan horizontal (Oxy) soit l'axe (Oy) . L'opération correspondante est une rotation d'angle $-\lambda$ autour de l'axe (Oz) . La matrice de cette opération est :

$$M_3 = \begin{pmatrix} \cos \lambda & \sin \lambda & 0 \\ -\sin \lambda & \cos \lambda & 0 \\ 0 & 0 & 1 \end{pmatrix}.$$



- Ensuite on redresse le plan à la verticale en modifiant la latitude φ (de façon à ce que le plan contienne l'axe (Oz)). Il s'agit d'effectuer une rotation d'angle $-\varphi$ autour de l'axe (Oy). La matrice de cette transformation est :

$$M_2 = \begin{pmatrix} \cos \varphi & 0 & -\sin \varphi \\ 0 & 1 & 0 \\ \sin \varphi & 0 & \cos \varphi \end{pmatrix}.$$

- Le plan de projection est devenu le plan (Oyz), il ne reste alors plus qu'à projeter suivant l'axe (Ox), la matrice de cette projection est la matrice M'_1 du cas simple.
- Les formules s'obtiennent par composition :

$$\begin{pmatrix} X \\ Y \end{pmatrix} = M \begin{pmatrix} x \\ y \\ z \end{pmatrix} \quad \text{où} \quad M = M'_1 M_2 M_3.$$

□

Remarque. Si on veut les coordonnées $\begin{pmatrix} x' \\ y' \\ z' \end{pmatrix}$ du projeté P' dans les coordonnées originales ($Oxyz$), les formules sont :

$$\begin{pmatrix} x' \\ y' \\ z' \end{pmatrix} = (M_2 M_3)^{-1} M_1 (M_2 M_3) \begin{pmatrix} x \\ y \\ z \end{pmatrix}.$$

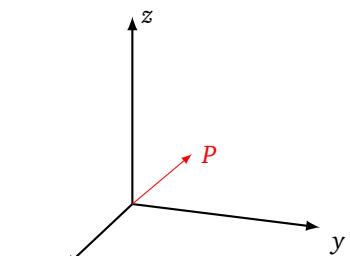
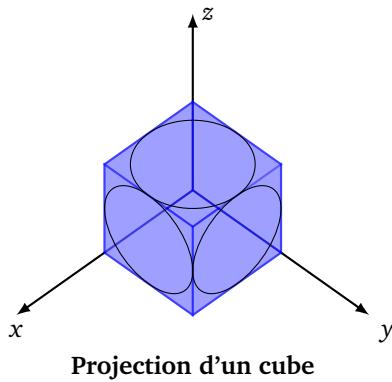
2.4. Cas particuliers

Des valeurs classiques sont :

- $\lambda = 30^\circ$ et $\varphi = 20^\circ$,
- $\lambda = \frac{\pi}{4} = 45^\circ$ et différentes valeurs de φ .

Exemple (Perspective isométrique).

Les paramètres sont $\lambda = \frac{\pi}{4} = 45^\circ$ et φ tel que $\sin \varphi = \frac{1}{\sqrt{3}}$ (de sorte que $\varphi \simeq 35^\circ$). Pour la perspective isométrique les trois axes ont la même importance (mais ce n'est pas une isométrie au sens géométrique usuel). Le vecteur normal au plan de projection est $\begin{pmatrix} 1 \\ 1 \\ 1 \end{pmatrix}$.



La matrice de cette projection est :

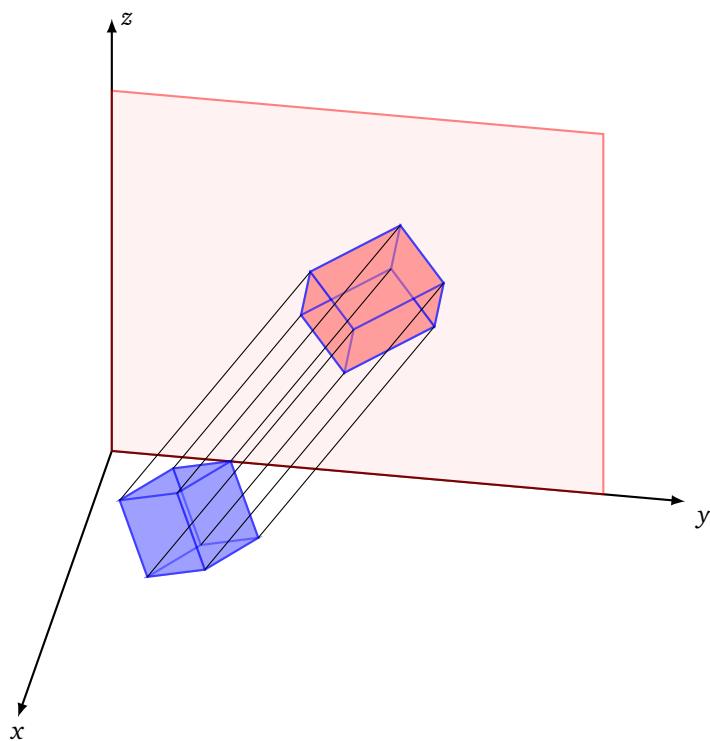
$$\begin{pmatrix} -\frac{\sqrt{2}}{2} & \frac{\sqrt{2}}{2} & 0 \\ \frac{\sqrt{2}}{2\sqrt{3}} & \frac{\sqrt{2}}{2\sqrt{3}} & \frac{\sqrt{2}}{\sqrt{3}} \end{pmatrix}$$

Les distances sur chacun des axes sont diminuées d'un coefficient multiplicatif $\cos \varphi = \frac{\sqrt{2}}{\sqrt{3}} \simeq 0.82$.

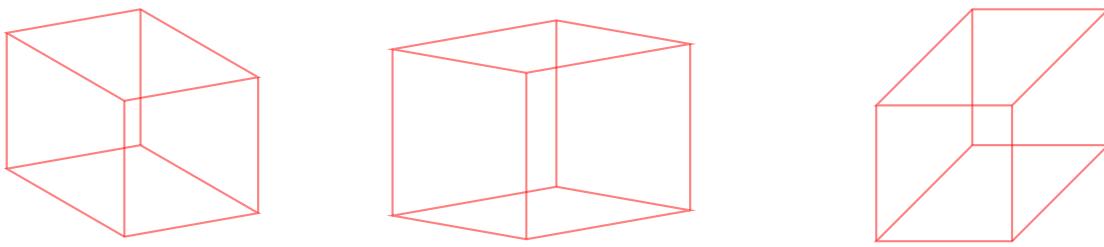
3. Projection parallèle

3.1. Principe

La **projection parallèle** est la généralisation de la projection précédente, sauf que l'axe de projection n'est plus supposé orthogonal au plan de projection.



Voici quelques projections parallèles d'un cube.



Cette projection s'appelle aussi ***perspective axonométrique***. Elle est utilisée dans le dessin technique, en infographie, en architecture... Deux objets identiques auront les mêmes dimensions qu'ils soient proches ou éloignés du plan de projection. Nous allons adopter un point de vue différent de celui de la projection orthogonale.

3.2. Données d'une axonométrie

Une projection parallèle est définie par :

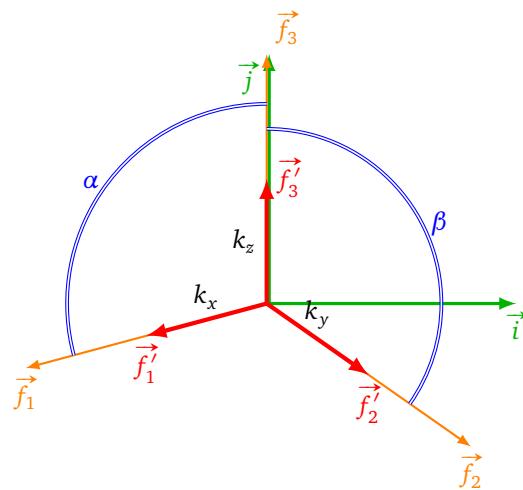
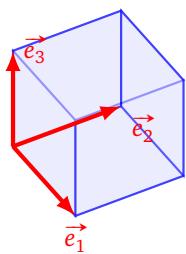
- la donnée de 3 angles α, β, γ (avec $\alpha + \beta + \gamma = 2\pi$),
- et 3 coefficients k_x, k_y, k_z .

Les angles déterminent 3 vecteurs unitaires $\vec{f}_1, \vec{f}_2, \vec{f}_3$ du plan :

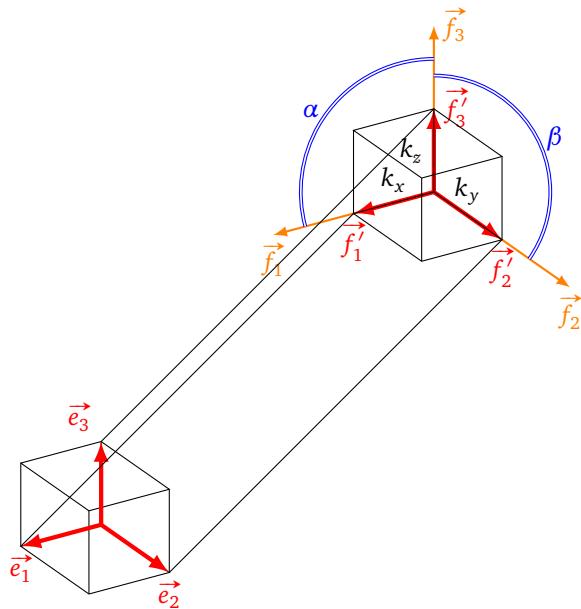
- \vec{f}_3 est par convention vertical dirigé vers le haut,
- \vec{f}_1 forme un angle $-\alpha$ avec \vec{f}_3 ,
- \vec{f}_2 forme un angle β avec \vec{f}_3 .

Les vecteurs $\vec{f}_1, \vec{f}_2, \vec{f}_3$ sont multipliés par par le facteur d'échelle correspondant à son axe :

- $\vec{f}'_1 = k_x \vec{f}_1$,
- $\vec{f}'_2 = k_y \vec{f}_2$,
- $\vec{f}'_3 = k_z \vec{f}_3$.



Si $(\vec{e}_1, \vec{e}_2, \vec{e}_3)$ est la base orthonormée canonique de \mathbb{R}^3 , alors la projection chaque vecteur \vec{e}_i se projette sur \vec{f}'_i .



3.3. Formule

Proposition 4.

La formule de la projection parallèle est

$$\begin{pmatrix} X \\ Y \\ Z \end{pmatrix} = M \begin{pmatrix} x \\ y \\ z \end{pmatrix} \quad \text{où} \quad M = \begin{pmatrix} -k_x \sin \alpha & k_y \sin \beta & 0 \\ k_x \cos \alpha & k_y \cos \beta & k_z \end{pmatrix}.$$

Autrement dit :

$$\begin{cases} X = -k_x \sin(\alpha) \cdot x + k_y \sin(\beta) \cdot y \\ Y = k_x \cos(\alpha) \cdot x + k_y \cos(\beta) \cdot y + k_z \cdot z \end{cases}$$

L'idée de ces formules est simple. Si $(\vec{e}_1, \vec{e}_2, \vec{e}_3)$ est la base orthonormée canonique de \mathbb{R}^3 , alors chaque vecteur \vec{e}_i se projette sur \vec{f}'_i qui est un multiple de \vec{f}_i :

- $\vec{e}_1 \mapsto \vec{f}'_1 = k_x \vec{f}_1$,
- $\vec{e}_2 \mapsto \vec{f}'_2 = k_y \vec{f}_2$,
- $\vec{e}_3 \mapsto \vec{f}'_3 = k_z \vec{f}_3$.

Ainsi trois vecteurs de l'espace s'envoient sur 3 vecteurs du plan.

Démonstration. Soit P un point de coordonnées $\begin{pmatrix} x \\ y \\ z \end{pmatrix}$, c'est-à-dire :

$$\overrightarrow{OP} = x\vec{e}_1 + y\vec{e}_2 + z\vec{e}_3.$$

La projection parallèle envoie \vec{e}_i sur \vec{f}'_i , donc l'image de \overrightarrow{OP} est :

$$xk_x\vec{f}_1 + yk_y\vec{f}_2 + zk_z\vec{f}_3.$$

Or :

$$\vec{f}_1 = \begin{pmatrix} -\sin \alpha \\ \cos \alpha \end{pmatrix} \quad \vec{f}_2 = \begin{pmatrix} \sin \beta \\ \cos \beta \end{pmatrix} \quad \vec{f}_3 = \begin{pmatrix} 0 \\ 1 \end{pmatrix}.$$

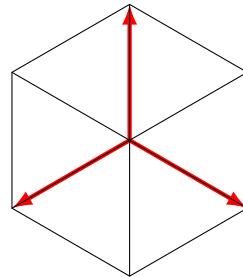
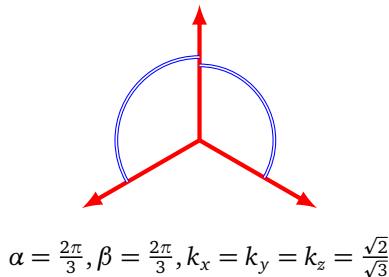
Ainsi la projection est l'application :

$$\begin{pmatrix} x \\ y \\ z \end{pmatrix} \longmapsto xk_x \begin{pmatrix} -\sin \alpha \\ \cos \alpha \end{pmatrix} + yk_y \begin{pmatrix} \sin \beta \\ \cos \beta \end{pmatrix} + zk_z \begin{pmatrix} 0 \\ 1 \end{pmatrix}.$$

D'où le résultat. □

3.4. Cas particuliers

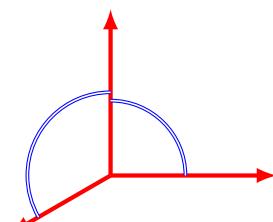
Perspective axonométrique. Cette projection est caractérisée par $\alpha = \beta = \frac{2\pi}{3}$ et $k_x = k_y = k_z = \frac{\sqrt{2}}{\sqrt{3}}$.



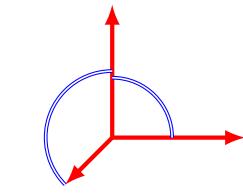
Perspective cavalière. Très utilisée en dessins techniques faits à la main. On impose $\beta = \frac{\pi}{2}$ et $k_y = k_z = 1$. Il reste deux paramètres α et k_x à fixer. Des valeurs classiques sont $\alpha = \frac{3\pi}{4}$ et $k_x = \frac{1}{2}$ ou $k_x = 1$.

La matrice de cette projection est :

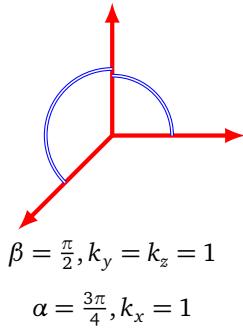
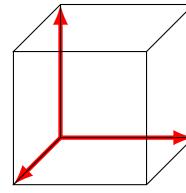
$$\begin{pmatrix} -k_x \sin \alpha & 1 & 0 \\ k_x \cos \alpha & 0 & 1 \end{pmatrix}$$



$$\alpha, \beta = \frac{\pi}{2}, k_x, k_y = k_z = 1$$

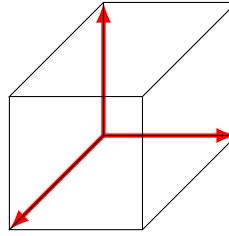


$$\begin{aligned} \beta &= \frac{\pi}{2}, k_y = k_z = 1 \\ \alpha &= \frac{3\pi}{4}, k_x = \frac{1}{2} \end{aligned}$$



$$\beta = \frac{\pi}{2}, k_y = k_z = 1$$

$$\alpha = \frac{3\pi}{4}, k_x = 1$$



3.5. Théorème de Pohlke

Le théorème de Pohlke (que nous admettrons) affirme que, quelles que soient nos données, il existe une projection parallèle correspondante.

Théorème 1 (Pohlke).

Tout triplet $(\vec{f}_1', \vec{f}_2', \vec{f}_3')$ qui engendre le plan \mathbb{R}^2 est l'image, à homothétie près, d'une base orthonormée $(\vec{e}_1, \vec{e}_2, \vec{e}_3)$ de l'espace \mathbb{R}^3 par une projection parallèle. Autrement dit, il existe un cube de l'espace (de longueur de côté k) dont les côtés de base sont $(k\vec{e}_1, k\vec{e}_2, k\vec{e}_3)$ qui se projette parallèlement sur $(\vec{f}_1', \vec{f}_2', \vec{f}_3')$.

L'unique condition de ce théorème est que les trois vecteurs $\vec{f}_1', \vec{f}_2', \vec{f}_3'$ ne soient pas colinéaires.

4. Autres perspectives

Nous étendons notre étude à des projections plus exotiques, dont certaines permettent d'avoir une vision large de la scène.

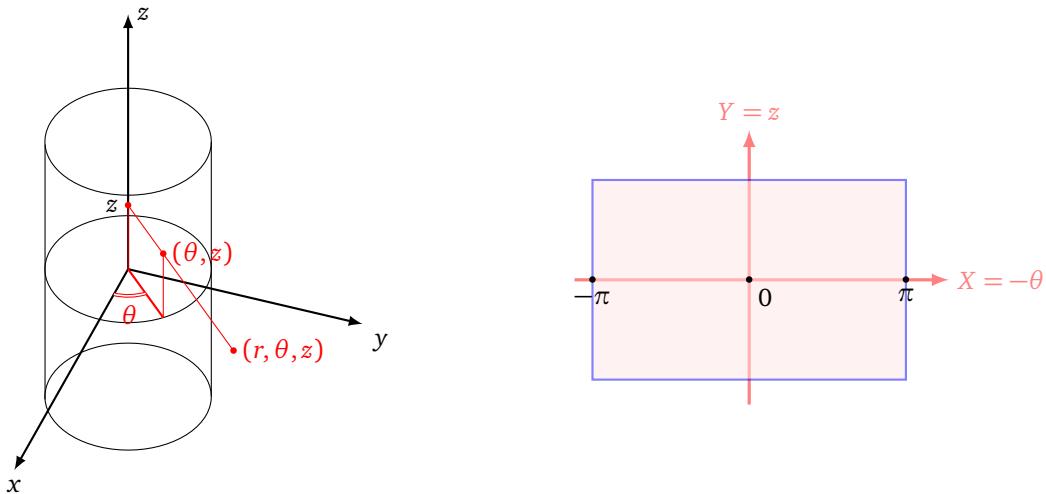
4.1. Coordonnées cylindriques

Expliquons comment les coordonnées cylindriques permettent d'avoir une vision à 360° . On imagine qu'on dispose d'un écran souple qu'on enroule pour former un cylindre.

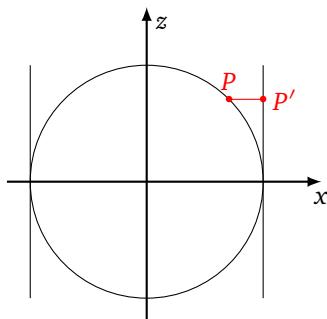
Il s'agit tout simplement de la transformation suivante $(x, y, z) \mapsto (-\theta, z)$:

$$P(x, y, z) \mapsto P(r, \theta, z) \mapsto P'(X, Y) = (-\theta, z)$$

On part des coordonnées cartésiennes que l'on transforme en coordonnées cylindriques, puis on projette sur le cylindre (on oublie la coordonnée r). L'angle θ (donc X) varie de $-\pi$ à π et z (donc Y) appartient à \mathbb{R} (mais dans la pratique est borné).



Cette projection cylindrique est du même type que la projection de Mercator pour représenter le globe terrestre à plat. Elle conduit à une forte distorsion au niveau des pôles.

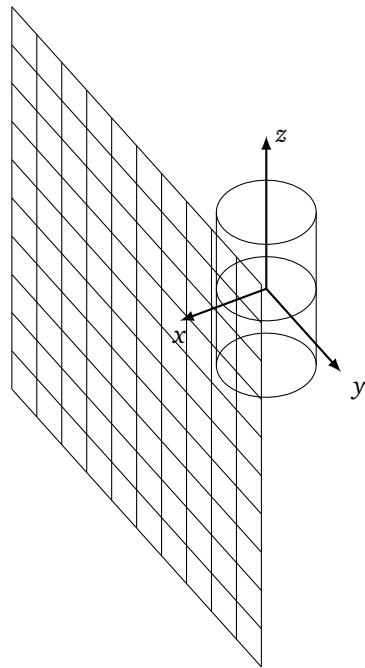


On peut bien sûr limiter la vision à 180° en se limitant au demi-cylindre défini par $-\frac{\pi}{2} \leq \theta \leq \frac{\pi}{2}$.

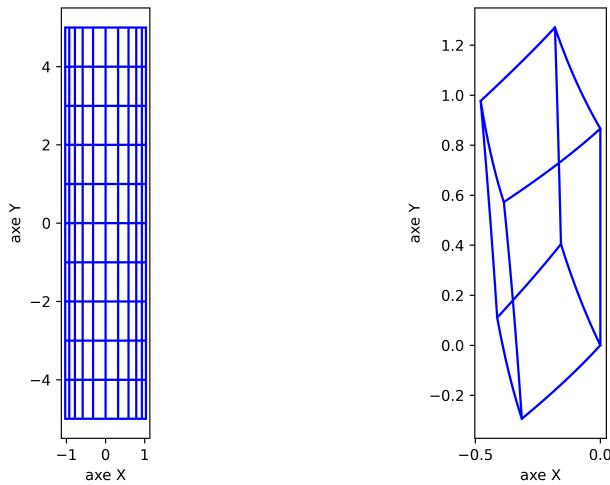
On rappelle les formules de passage $(x, y, z) \mapsto (r, \theta, z)$:

$$\begin{cases} r &= \sqrt{x^2 + y^2} \\ \theta &= \arctan2(y, x) \end{cases}$$

Par exemple prenons une grille dans le plan ($x = 3$) :



Voici sa projection cylindrique ainsi que celle d'un cube (on pourrait ajouter un agrandissement le long de l'axe X pour améliorer le rendu).

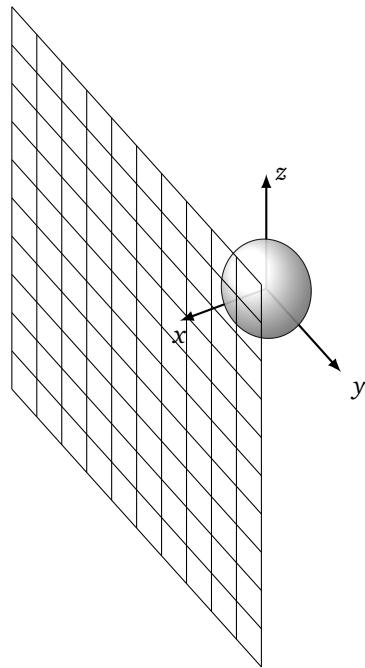


4.2. Coordonnées sphériques

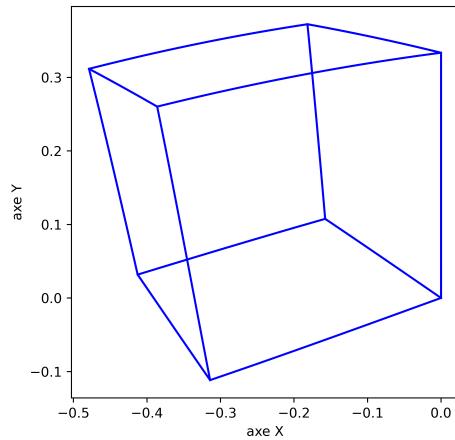
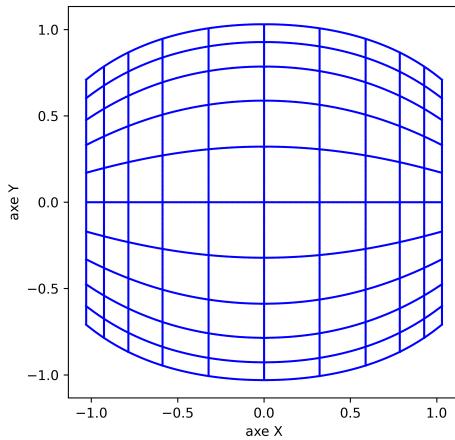
Remarque. On peut faire la même chose avec les coordonnées sphériques :

$$P(x, y, z) \longleftrightarrow P(r, \varphi, \lambda) \longleftrightarrow P'(X, Y) = (-\lambda, \varphi)$$

où $\varphi \in [-\frac{\pi}{2}, \frac{\pi}{2}]$ est la latitude et $\lambda \in]-\pi, \pi]$ est la longitude.



On pourrait imaginer beaucoup d'autres variantes, mais comme pour les projections cartographiques, aucune n'est parfaitement fidèle.

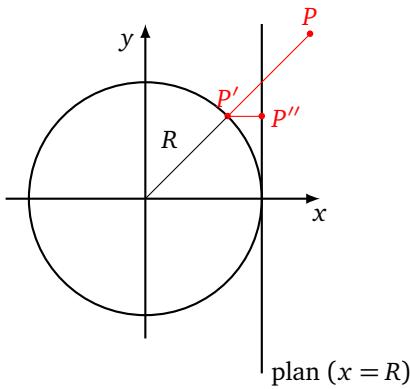


4.3. Perspective curviligne cylindrique

La perspective curviligne cylindrique part du principe que dans une vision à 180° les objets sur la droite et la gauche du champ de vision devraient paraître plus petits que les objets en face.

On considère un cylindre vertical, d'axe (Oz), de rayon R . La projection s'effectue sur le plan vertical \mathcal{P} d'équation ($x = R$) tangent au cylindre. La construction est la suivante :

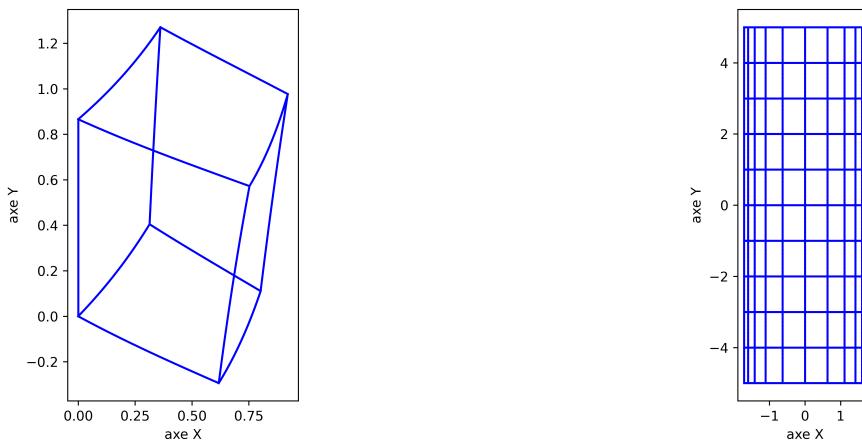
- On part d'un point P de l'espace,
- on le projette sur le point P' du cylindre,
- puis on projette ce point orthogonalement sur le plan \mathcal{P} en un point P'' .



Notons $P(x, y, z)$ les coordonnées du point à projeter et $r = \sqrt{x^2 + y^2}$. La transformation est la suivante :

$$P(x, y, z) \longmapsto P'\left(x \frac{R}{r}, y \frac{R}{r}, z\right) \longmapsto P''(X, Y) = \left(y \frac{R}{r}, z\right)$$

Cette transformation conserve la hauteur z des points.



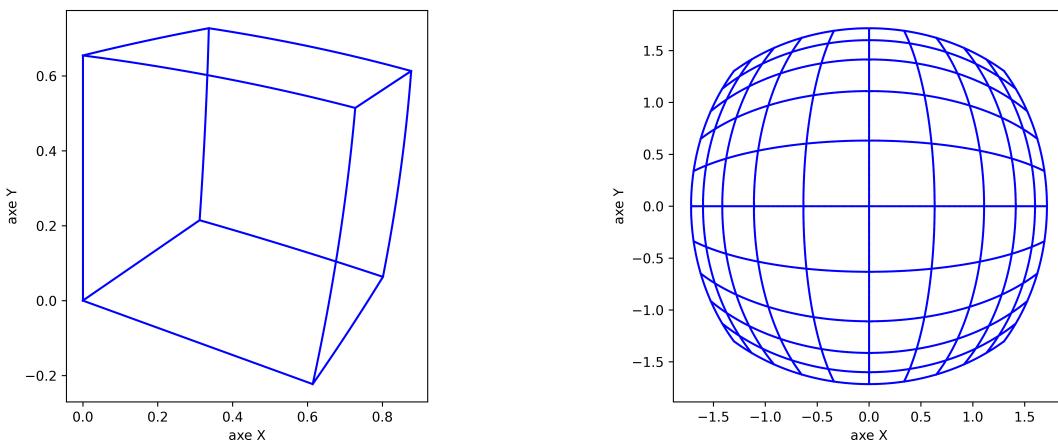
4.4. Perspective curviligne sphérique

On étend la construction précédente sur le même principe avec une distorsion haut/bas, en plus de la distorsion gauche/droite. On obtient une vue à 180° de gauche à droite, mais aussi de haut en bas. Le résultat obtenu ressemble à une photographie prise avec un objectif de type *fisheye*.

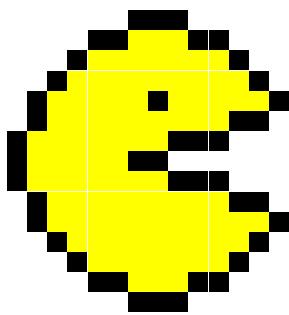
Dans cette construction, on projette le point sur une (demi-)sphère de rayon R , puis sur un plan vertical tangent.

Pour $P(x, y, z)$ on note cette fois $r = \sqrt{x^2 + y^2 + z^2}$. La transformation est :

$$P(x, y, z) \longmapsto P'\left(x \frac{R}{r}, y \frac{R}{r}, z \frac{R}{r}\right) \longmapsto P''(X, Y) = \left(y \frac{R}{r}, z \frac{R}{r}\right)$$



DEUXIÈME PARTIE



IMAGES

Lancer de rayons I

Nous abordons les bases du ray-tracing : nous lançons un rayon depuis une source et calculons en quel(s) point(s) ce rayon atteint un objet géométrique (plan, triangle, sphère...).

Le principe général et l'utilisation du ray-tracing seront expliqués plus tard dans un chapitre plus avancé.

1. Intersection avec une surface plane

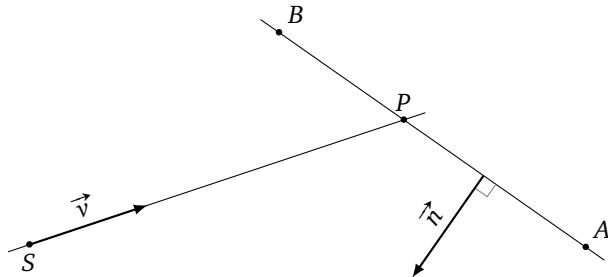
Voici les notations utilisées pour ce chapitre :

- Point S : la source, c'est l'origine du rayon. Contrairement à ce que l'on peut penser cela ne correspondra pas à la source lumineuse, mais à la position de l'observateur ou celle de la caméra.
- Vecteur \vec{v} : direction du rayon. On pourra normaliser \vec{v} pour simplifier certains calculs.
- $S + t\vec{v}$ ($t \in \mathbb{R}$) : paramétrisation du rayon (issu de S , dirigé par \vec{v}).
- Point P : intersection de l'objet étudié avec le rayon (ou P_1, P_2, \dots en cas d'intersections multiples).

1.1. Dans le plan : intersection avec un segment

À titre d'illustration et de motivation commençons par étudier le problème dans le plan.

- L'objet étudié est un segment $[AB]$ où A et B sont deux points distincts du plan.
- On se donne un rayon défini par un point S et un vecteur \vec{v} .
- Question 1. Le rayon coupe-t-il la droite (AB) ? le segment $[AB]$?
- Question 2. Si oui, quel est ce point d'intersection ?



Proposition 1.

- Le rayon coupe la droite (AB) si et seulement si $\vec{v} \cdot \vec{n} \neq 0$ où \vec{n} est un vecteur (non nul) orthogonal à (AB) .
- Le point d'intersection est alors $P = S + t\vec{v}$ avec

$$t = \frac{\vec{SA} \cdot \vec{n}}{\vec{v} \cdot \vec{n}}$$

3. Il s'écrit aussi $P = A + \alpha \vec{AB}$ avec

$$\alpha = -\frac{\overrightarrow{SA} \cdot \vec{m}}{\overrightarrow{AB} \cdot \vec{m}}$$

où \vec{m} est un vecteur (non nul) orthogonal à \vec{v} .

4. Le point d'intersection P appartient au segment $[AB]$ si et seulement si $0 \leq \alpha \leq 1$.

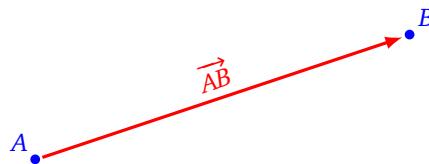
On détaille la preuve ci-dessous.

Démonstration.

L'intersection avec la droite existe-t-elle ? On décide facilement si le rayon coupe la droite : le rayon intersecte la droite (AB) si et seulement si \vec{AB} et \vec{v} ne sont pas parallèles. Ainsi, si on note \vec{n} un vecteur (non nul) orthogonal à (AB) , l'intersection existe si et seulement si $\vec{v} \cdot \vec{n} \neq 0$.

Intersection. Pour décider si le rayon coupe le segment, on a besoin de calculer l'intersection du rayon avec la droite (AB) . Le point P d'intersection est un point du rayon, donc $P = S + t \vec{v}$ pour un certain $t \in \mathbb{R}$. Le point P appartient à la droite (AB) donc s'écrit $P = A + \alpha \vec{AB}$ pour un certain $\alpha \in \mathbb{R}$.

Petits rappels sur les notations affines : $P = S + t \vec{v}$ signifie que P est le point vérifiant $\overrightarrow{SP} = t \vec{v}$, de même $P = A + \alpha \vec{AB}$ signifie $\overrightarrow{AP} = \alpha \vec{AB}$. Enfin la différence de deux points $B - A$ est le vecteur \vec{AB} .



Le point P est l'intersection du rayon et de la droite, ainsi t et α vérifient l'égalité :

$$S + t \vec{v} = A + \alpha \vec{AB}.$$

Autrement dit :

$$t \vec{v} = A - S + \alpha \vec{AB}.$$

Ce qui donne :

$$t \vec{v} = \overrightarrow{SA} + \alpha \vec{AB} \quad (1)$$

Calcul de t . On prend le produit scalaire avec \vec{n} de part et d'autre de l'égalité (1) :

$$t \vec{v} \cdot \vec{n} = \overrightarrow{SA} \cdot \vec{n} + \alpha \vec{AB} \cdot \vec{n}$$

Comme \vec{n} est orthogonal à \vec{AB} :

$$t \vec{v} \cdot \vec{n} = \overrightarrow{SA} \cdot \vec{n} + 0,$$

ainsi

$$t = \frac{\overrightarrow{SA} \cdot \vec{n}}{\vec{v} \cdot \vec{n}}.$$

Cela suffit pour calculer l'intersection du rayon avec la droite (AB) . Mais pour le segment $[AB]$ il faut calculer α .

Calcul de α . On repart de l'égalité (1), mais cette fois on effectue le produit scalaire avec \vec{m} , vecteur orthogonal à \vec{v} :

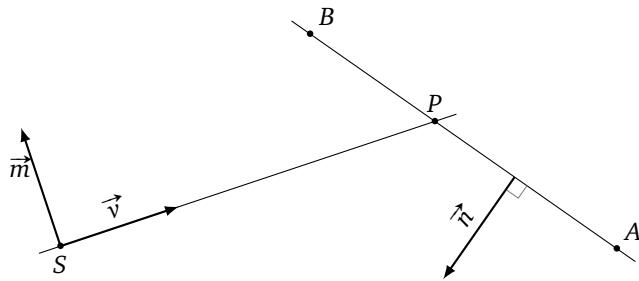
$$t \vec{v} \cdot \vec{m} = \overrightarrow{SA} \cdot \vec{m} + \alpha \vec{AB} \cdot \vec{m},$$

donc

$$0 = \overrightarrow{SA} \cdot \vec{m} + \alpha \vec{AB} \cdot \vec{m}.$$

D'où :

$$\alpha = -\frac{\overrightarrow{SA} \cdot \vec{m}}{\vec{AB} \cdot \vec{m}}.$$



Appartenance au segment $[AB]$. Le point P appartient au segment $[AB]$ si et seulement si $\alpha \geq 0$ et $\alpha \leq 1$. (Si $\alpha < 0$ le point P est « avant » A , si $\alpha > 1$ le point P est « après » B .)

□

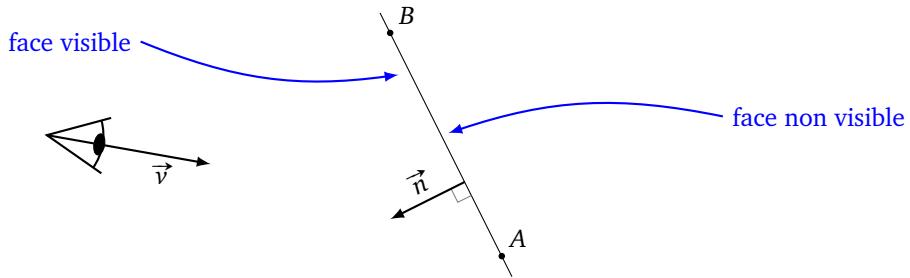
On peut apporter quelques précisions :

- *L'objet est-il devant ou derrière la source ?*

Le rayon coupe l'objet à un temps positif si et seulement si $t \geq 0$. Sinon le rayon coupe l'objet « dans le passé », c'est-à-dire que l'objet est situé derrière la source, ce que l'on ne considère généralement pas comme valide.

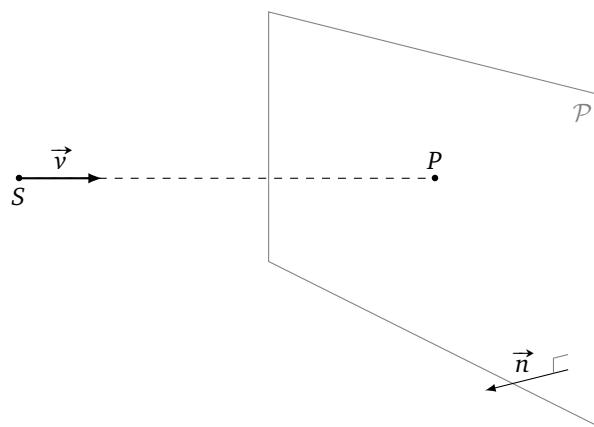
- *Le rayon coupe-t-il l'objet sur sa face visible ?*

Désignons (arbitrairement) la face visible par le vecteur \vec{n} sortant, de sorte que $(\overrightarrow{AB}, \vec{n})$ soit une base directe du plan. Le rayon éclaire la face visible si et seulement si $\vec{v} \cdot \vec{n} < 0$.



1.2. Plan d'équation $ax + by + cz + d = 0$

Revenons dans l'espace : lançons un rayon vers un plan et calculons le point d'intersection.



Soit \mathcal{P} un plan d'équation $ax + by + cz + d = 0$. Le vecteur $\vec{n} = \begin{pmatrix} a \\ b \\ c \end{pmatrix}$ est un vecteur orthogonal au plan.

On note $S = \begin{pmatrix} x_S \\ y_S \\ z_S \end{pmatrix}$ et $\vec{v} = \begin{pmatrix} x_v \\ y_v \\ z_v \end{pmatrix}$.

Proposition 2.

Le rayon intersecte le plan si et seulement si $\vec{v} \cdot \vec{n} \neq 0$. Si c'est le cas le point d'intersection est $P = S + t \vec{v}$ avec :

$$t = -\frac{ax_S + by_S + cz_S + d}{ax_v + by_v + cz_v}$$

Si on note $O = \begin{pmatrix} 0 \\ 0 \\ 0 \end{pmatrix}$ l'origine du repère, alors on peut récrire :

$$t = -\frac{\overrightarrow{OS} \cdot \vec{n} + d}{\vec{v} \cdot \vec{n}}$$

Démonstration. Le rayon ne coupe pas le plan lorsqu'il est parallèle au plan, c'est-à-dire lorsque \vec{v} et \vec{n} sont orthogonaux. Ainsi l'intersection existe si et seulement $\vec{v} \cdot \vec{n} \neq 0$.

Les coordonnées du point P sont :

$$P = S + t \vec{v} = \begin{pmatrix} x_S \\ y_S \\ z_S \end{pmatrix} + t \begin{pmatrix} x_v \\ y_v \\ z_v \end{pmatrix} = \begin{pmatrix} x_S + tx_v \\ y_S + ty_v \\ z_S + tz_v \end{pmatrix}.$$

Le point P appartient au plan \mathcal{P} si ses coordonnées vérifient l'équation $ax + by + cz + d = 0$, donc :

$$\begin{aligned} & a(x_S + tx_v) + b(y_S + ty_v) + c(z_S + tz_v) + d = 0 \\ \iff & ax_S + by_S + cz_S + d + t(ax_v + by_v + cz_v) = 0 \\ \iff & t = -\frac{ax_S + by_S + cz_S + d}{ax_v + by_v + cz_v} \end{aligned}$$

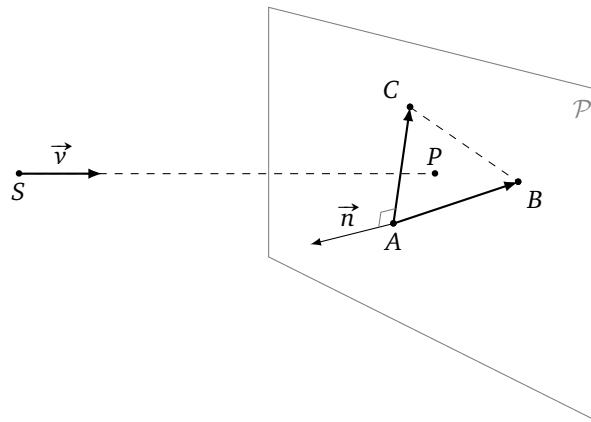
Le numérateur s'écrit aussi $\overrightarrow{OS} \cdot \vec{n} + d$ et le dénominateur $\vec{v} \cdot \vec{n}$.

□

1.3. Triangle

Considérons un triangle de l'espace défini par trois points A, B, C non alignés. On considère un vecteur \vec{n} orthogonal au triangle, nous choisissons :

$$\vec{n} = \overrightarrow{AB} \wedge \overrightarrow{AC}.$$



Le point P est sur le rayon, donc il existe $t \in \mathbb{R}$ tel que :

$$P = S + t \vec{v}$$

et P est dans le plan (ABC) donc il existe $\alpha, \beta \in \mathbb{R}$ tels que :

$$P = A + \alpha \overrightarrow{AB} + \beta \overrightarrow{AC}.$$

Proposition 3.

1. Le point d'intersection P entre le rayon et le plan (ABC) vérifie :

$$\boxed{t = \frac{\vec{SA} \cdot \vec{n}}{\vec{v} \cdot \vec{n}} \quad \alpha = -\frac{(\vec{SA} \wedge \vec{AC}) \cdot \vec{v}}{\vec{v} \cdot \vec{n}} \quad \beta = \frac{(\vec{SA} \wedge \vec{AB}) \cdot \vec{n}}{\vec{v} \cdot \vec{n}}}$$

2. Le point P appartient au triangle ABC si et seulement si :

$$\boxed{\alpha \geq 0 \quad \beta \geq 0 \quad \alpha + \beta \leq 1}$$

3. En plus, le triangle est devant la source si et seulement si $t \geq 0$ et intersecte le triangle sur sa face visible si et seulement si $\vec{v} \cdot \vec{n} < 0$.

Démonstration. Des équations

$$P = S + t \vec{v} \quad \text{et} \quad P = A + \alpha \vec{AB} + \beta \vec{AC},$$

on tire :

$$S + t \vec{v} = A + \alpha \vec{AB} + \beta \vec{AC},$$

d'où :

$$t \vec{v} = \vec{SA} + \alpha \vec{AB} + \beta \vec{AC} \tag{2}$$

Le produit scalaire avec \vec{n} à gauche et à droite de l'égalité (2) donne :

$$t \vec{v} \cdot \vec{n} = \vec{SA} \cdot \vec{n} + \alpha \vec{AB} \cdot \vec{n} + \beta \vec{AC} \cdot \vec{n}$$

Mais \vec{n} est orthogonal à \vec{AB} et à \vec{AC} , donc :

$$t \vec{v} \cdot \vec{n} = \vec{SA} \cdot \vec{n} + 0 + 0,$$

ce qui donne la valeur voulue pour t .

On repart de l'égalité (2) mais cette fois on effectue le produit vectoriel avec \vec{AC} sur le terme de gauche et celui de droite :

$$t \vec{v} \wedge \vec{AC} = \vec{SA} \wedge \vec{AC} + \alpha \vec{AB} \wedge \vec{AC} + \beta \vec{AC} \wedge \vec{AC}.$$

Par définition $\vec{AB} \wedge \vec{AC} = \vec{n}$ et on a $\vec{AC} \wedge \vec{AC} = \vec{0}$, d'où :

$$t \vec{v} \wedge \vec{AC} = \vec{SA} \wedge \vec{AC} + \alpha \vec{n}.$$

On effectue le produit scalaire par \vec{v} de part et d'autre :

$$t (\vec{v} \wedge \vec{AC}) \cdot \vec{v} = (\vec{SA} \wedge \vec{AC}) \cdot \vec{v} + \alpha \vec{n} \cdot \vec{v}.$$

Mais $\vec{v} \wedge \vec{AC}$ est orthogonal à \vec{v} donc $(\vec{v} \wedge \vec{AC}) \cdot \vec{v} = 0$, ce qui donne :

$$0 = (\vec{SA} \wedge \vec{AC}) \cdot \vec{v} + \alpha \vec{n} \cdot \vec{v},$$

d'où la valeur de α .

Pour trouver β , on repart de (2) en effectue le produit vectoriel avec \vec{AB} , puis le produit scalaire \vec{v} pour obtenir :

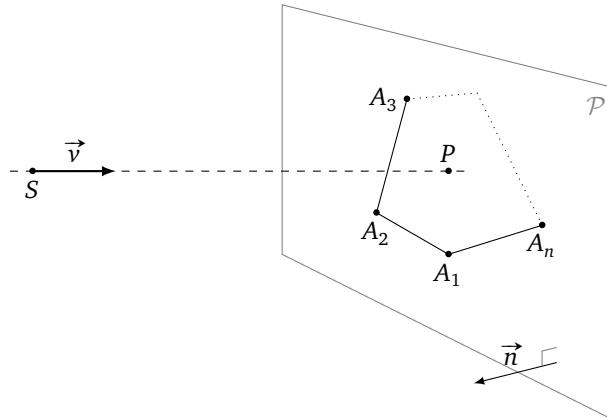
$$0 = (\vec{SA} \wedge \vec{AB}) \cdot \vec{v} - \beta \vec{n} \cdot \vec{v}.$$

□

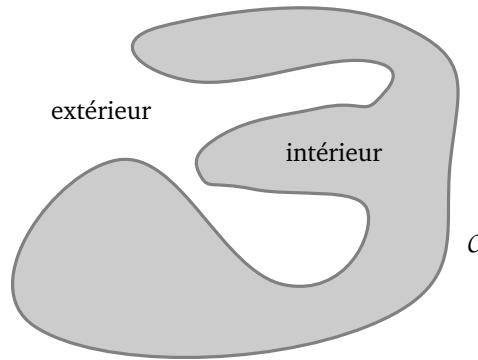
Autre méthode. Nous verrons une méthode plus efficace dans le chapitre « Lancer de rayons II ».

1.4. Polygone

Pour le lancer de rayon vers un polygone, il suffit de déterminer l'intersection du rayon avec le plan contenant le polygone, puis de décider si le point d'intersection est à l'intérieur ou à l'extérieur du polygone.

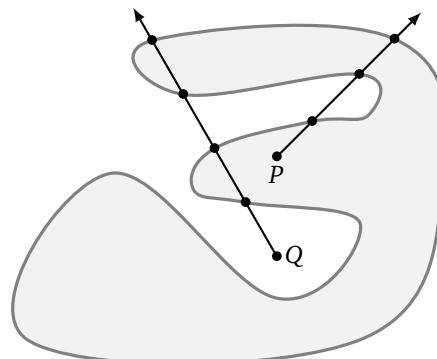


Tout d'abord la notion de **courbe de Jordan** permet d'identifier clairement l'intérieur de l'extérieur. Soit \mathcal{C} une courbe du plan, continue, fermée (elle se referme sur elle-même), simple (elle ne se recoupe pas), alors \mathcal{C} découpe le plan en deux régions : une partie compacte (l'**intérieur**) et une partie non compacte (l'**extérieur**). L'intérieur et l'extérieur s'intersectent exactement en \mathcal{C} .



Nous allons voir trois méthodes pour décider si un point P est à l'intérieur ou l'extérieur d'un polygone. (On ne discutera pas du cas particulier où P est exactement sur le bord du polygone.) Les deux premières méthodes sont des méthodes planes.

A. Méthode topologique. On considère une courbe de Jordan \mathcal{C} du plan. Soit P un point du plan. On trace une demi-droite \mathcal{D}_+ issue de P . On compte le nombre N de fois où la demi-droite \mathcal{D}_+ intersecte la courbe \mathcal{C} . Le point P est à l'intérieur de \mathcal{C} si et seulement si N est impair.



Dans toute la suite on considère des polygones convexes.

B. Calculs dans le plan.

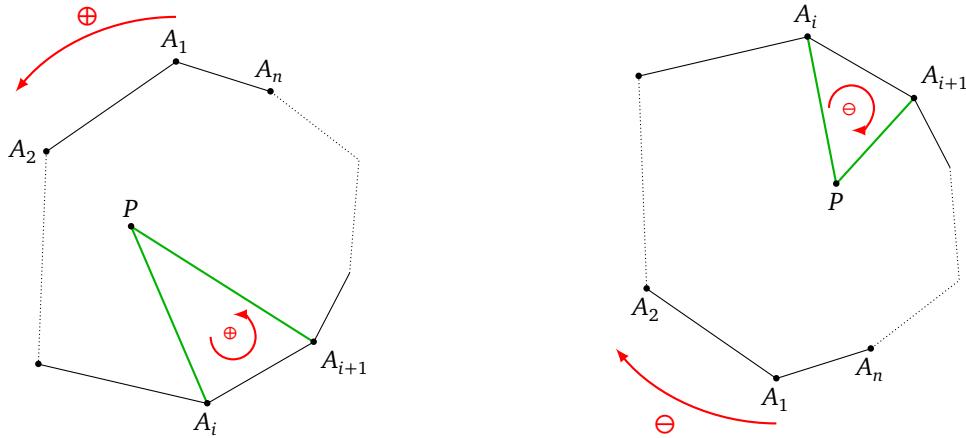
On considère un polygone convexe \mathcal{C} dans un repère orthonormé dont les sommets ordonnés sont $A_i(x_i, y_i)$, $i = 1, \dots, n$ dans un plan \mathcal{P} . On pose $A_{n+1} = A_1$. La numérotation peut être dans le sens trigonométrique positif ou négatif.

Proposition 4.

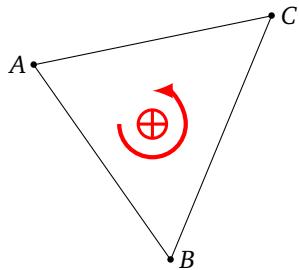
Le point $P(x, y)$ est à l'intérieur du polygone \mathcal{C} si et seulement si les nombres :

$$\det(\overrightarrow{PA_i}, \overrightarrow{PA_{i+1}}) = (x - x_i)(y - y_{i+1}) - (x - x_{i+1})(y - y_i) \quad i = 1, \dots, n$$

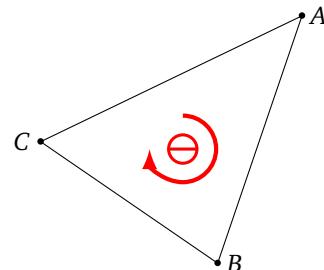
sont tous de même signe (tous positifs ou tous négatifs).



Avant la preuve, distinguons un triangle ABC **positif** (A, B, C apparaissent dans l'ordre trigonométrique) d'un triangle ABC **négatif**.



Triangle ABC négatif



Triangle ABC positif

Démonstration. Considérons le cas d'un polygone numéroté positivement. Pour un point P à l'intérieur du polygone, tout triangle PA_iA_{i+1} est positif, quel que soit $i = 1, \dots, n$. Alors que pour un point P situé à l'extérieur certains triangles PA_iA_{i+1} sont positifs et d'autres PA_jA_{j+1} sont négatifs.

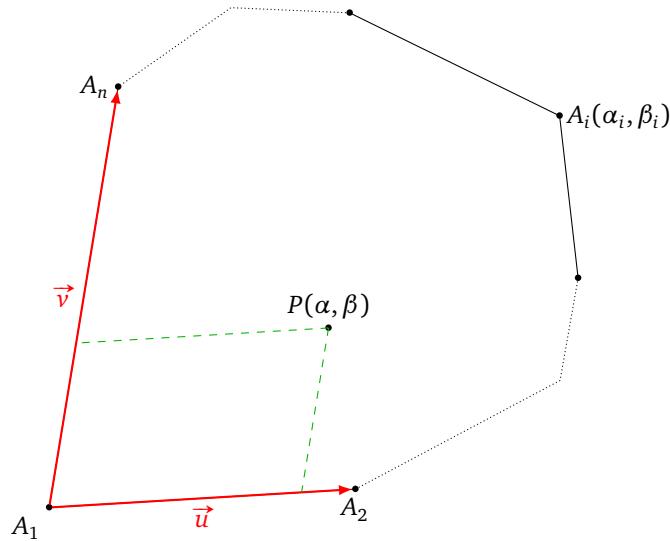
Or :

$$\begin{aligned} & \text{Le triangle } PA_iA_{i+1} \text{ est positif} \\ \iff & \det(\overrightarrow{PA_i}, \overrightarrow{PA_{i+1}}) \geq 0 \\ \iff & \begin{vmatrix} x - x_i & x - x_{i+1} \\ y - y_i & y - y_{i+1} \end{vmatrix} \geq 0 \\ \iff & (x - x_i)(y - y_{i+1}) - (x - x_{i+1})(y - y_i) \geq 0 \end{aligned}$$

D'où le résultat dans le cas où la numérotation est orientée positivement. Le cas d'une numérotation orientée négativement est similaire. \square

Le résultat précédent n'est pas nécessairement très pratique car il nécessite de calculer les coordonnées des sommets A_i du polygone dans un repère orthonormé du plan \mathcal{P} contenant le polygone, mais souvenons-nous que ce plan \mathcal{P} est un plan de l'espace.

Il peut être plus facile d'effectuer les calculs dans un autre système de coordonnées. Considérons le repère (A_1, \vec{u}, \vec{v}) d'origine A_1 et de base définie par $\vec{u} = \overrightarrow{A_1 A_2}$ et $\vec{v} = \overrightarrow{A_1 A_n}$. Dans ce repère les coordonnées de A_1 sont $(0, 0)$, celles de A_2 sont $(1, 0)$, celles de A_n sont $(0, 1)$. De façon générale on note $A_i(\alpha_i, \beta_i)$ les coordonnées des sommets et $P(\alpha, \beta)$ les coordonnées du point P dans ce repère.



On a le même résultat que précédemment dans ce nouveau repère.

Proposition 5.

Le point $P(\alpha, \beta)$ est à l'intérieur du polygone C si et seulement si les nombres :

$$(\alpha - \alpha_i)(\beta - \beta_{i+1}) - (\alpha - \alpha_{i+1})(\beta - \beta_i) \quad i = 1, \dots, n$$

sont tous de même signe.

Le résultat est une conséquence de la proposition 4 après un changement de coordonnées

$$\begin{pmatrix} \alpha \\ \beta \end{pmatrix} = A \begin{pmatrix} x \\ y \end{pmatrix}$$

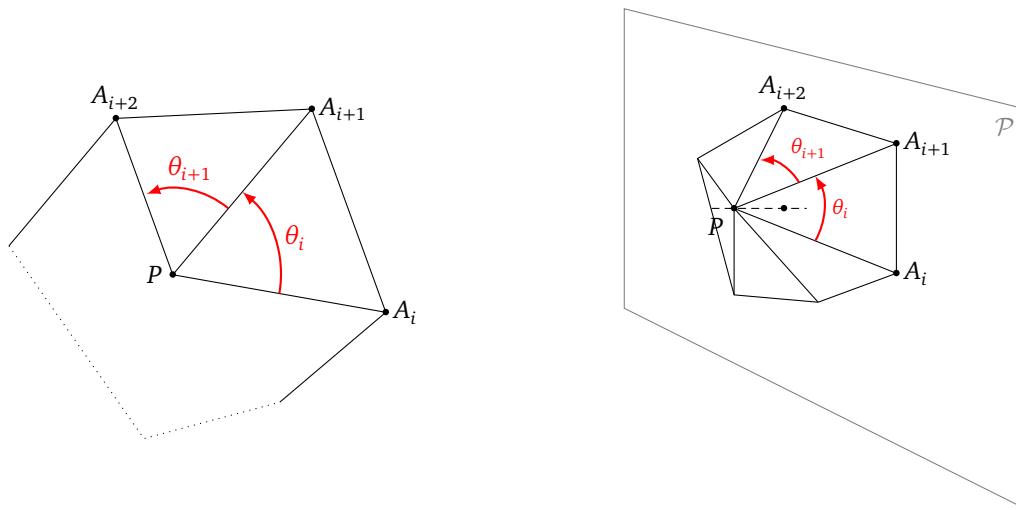
pour une certaine matrice A correspondant au changement de la base orthonormée (\vec{i}, \vec{j}) vers la base (\vec{u}, \vec{v}) .

C. Somme des angles.

Voici une méthode qui ne nécessite pas de ramener les calculs à un plan. Soit C un polygone de sommets A_1, \dots, A_n . Soit P un point du plan \mathcal{P} contenant le polygone. Notons θ_i l'angle (orienté) formé par $(\overrightarrow{PA_i}, \overrightarrow{PA_{i+1}})$.

Proposition 6.

Si P est à l'intérieur du polygone $\sum_{i=1}^n \theta_i = 2\pi$ (ou -2π). Si P est à l'extérieur du polygone $\sum_{i=1}^n \theta_i = 0$.



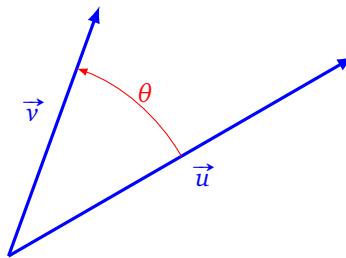
Rappelons comment calculer l'angle (non-orienté) $|\theta|$ entre deux vecteurs non nuls \vec{u} et \vec{v} . La relation

$$\vec{u} \cdot \vec{v} = \|\vec{u}\| \times \|\vec{v}\| \cos(\theta)$$

implique :

$$|\theta| = \arccos\left(\frac{\vec{u} \cdot \vec{v}}{\|\vec{u}\| \times \|\vec{v}\|}\right).$$

Cette formule est valable pour des vecteurs du plan, mais aussi pour des vecteurs de l'espace.



Par contre il faut expliquer comment attribuer (arbitrairement) un signe à θ .

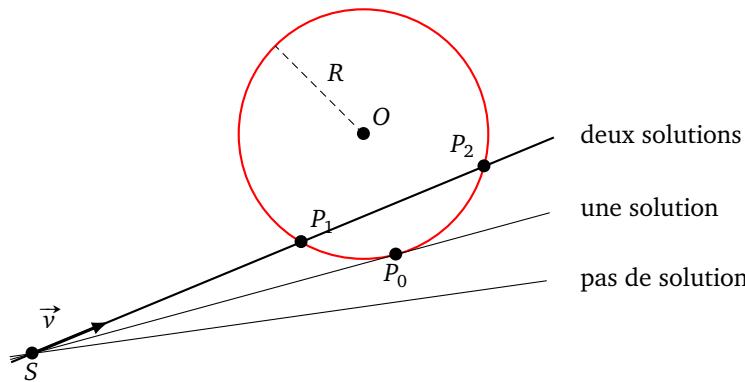
On fixe un vecteur \vec{n} orthogonal au plan \mathcal{P} du polygone (par exemple $\vec{n} = \overrightarrow{A_1A_2} \wedge \overrightarrow{A_1A_n}$). Pour chaque i on calcule le produit vectoriel $\vec{w}_i = \overrightarrow{PA_i} \wedge \overrightarrow{PA_{i+1}}$, puis $\vec{w}_i \cdot \vec{n}$. Le signe de ce produit scalaire est le signe de θ_i .

Un avantage de cette formule est sa stabilité. En effet, même si P n'est pas parfaitement dans le plan \mathcal{P} du polygone (par exemple juste au-dessus du plan ou juste en dessous) on peut encore décider si P est plutôt à l'intérieur ou plutôt à l'extérieur. Pour un point P plutôt à l'intérieur du polygone alors $\sum_{i=1}^n \theta_i \simeq 2\pi$ (en fait la somme sera proche mais strictement inférieure à 2π) ou $\sum_{i=1}^n \theta_i \simeq -2\pi$, pour un point plutôt à l'extérieur $\sum_{i=1}^n \theta_i \simeq 0$. Les alternatives possibles sont suffisamment différentes pour autoriser des incertitudes numériques sur les coordonnées et les calculs.

2. Intersection avec une sphère

2.1. Dans le plan : intersection avec un cercle

Considérons la situation plus simple du lancer d'un rayon dans le plan. Ce rayon intersecte-t-il un cercle donné et, si oui, en quels points ? Il y a 0 ou 2 solutions (et une seule dans le cas tangent).



Voici deux façons d'aborder le problème, que l'on retrouvera et détaillera dans le cas général juste après.

Méthode algébrique. Considérons le cercle \mathcal{C} de centre $O = (x_O, y_O)$ et de rayon R . Une équation est

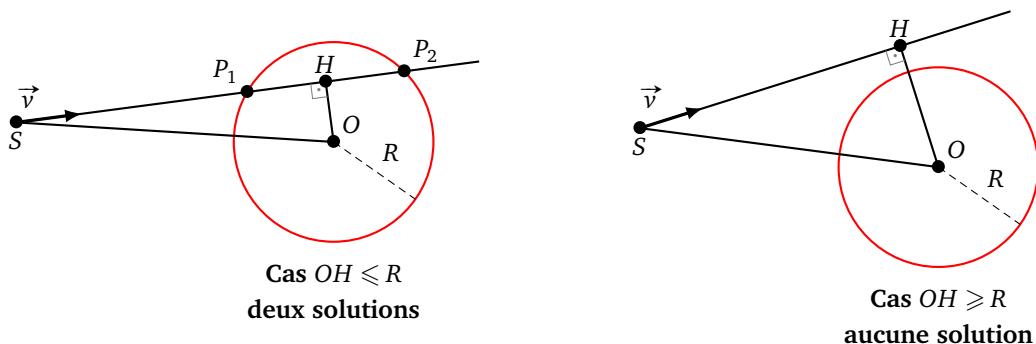
$$(x - x_O)^2 + (y - y_O)^2 = R^2.$$

Le paramétrage du point $P = (x, y)$ du rayon donne :

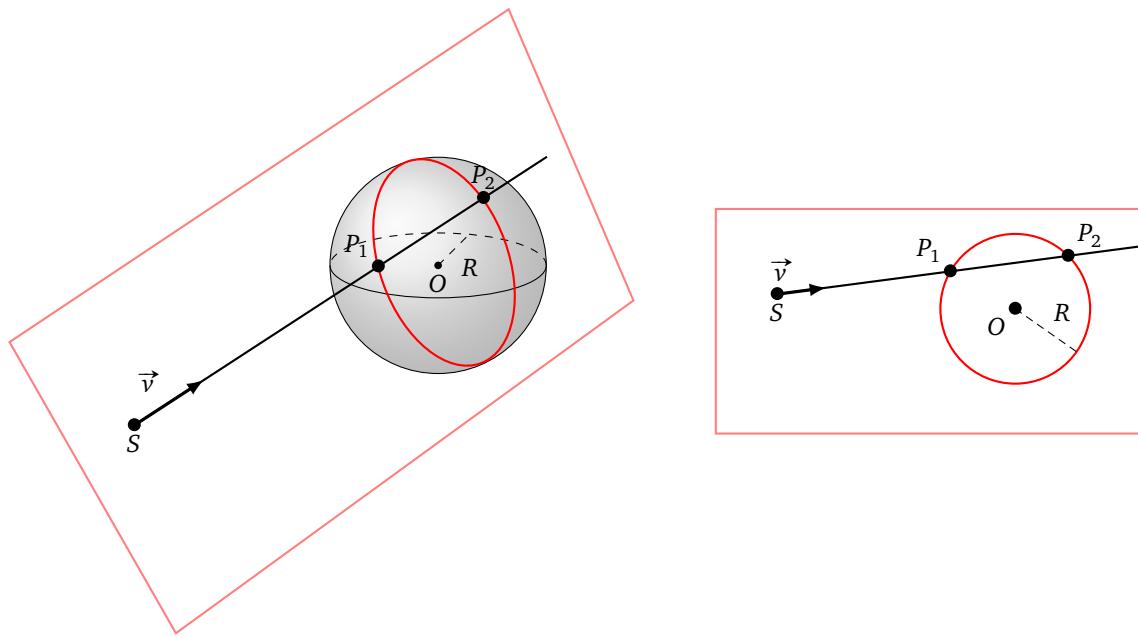
$$\begin{pmatrix} x \\ y \end{pmatrix} = \begin{pmatrix} x_S \\ y_S \end{pmatrix} + t \begin{pmatrix} x_v \\ y_v \end{pmatrix}.$$

On obtient donc 3 équations et 3 inconnues (t, x, y) . Mais attention ce n'est pas un système linéaire, car l'équation d'un cercle est de degré 2. Cependant nous arriverons à résoudre ce système et à exprimer les deux solutions P_1 et P_2 lorsque le rayon intersecte le cercle.

Méthode géométrique. Notons H le projeté orthogonal du centre O sur le rayon. Si $OH \leq R$ alors le rayon intersecte le cercle et on déterminera les deux solutions P_1 et P_2 (symétriques de part et d'autre de H).



Dans la suite on considère le lancer d'un rayon en direction d'une sphère \mathcal{S} de centre O et de rayon R . On peut noter que si on considère le plan \mathcal{P} qui contient le rayon et le centre de la sphère alors l'intersection de \mathcal{P} avec \mathcal{S} est un cercle \mathcal{C} de centre O et de rayon R ; on ramène ainsi le problème dans l'espace à un problème du plan.



2.2. Sphère : formule algébrique

Considérons la sphère S de centre $O = (x_O, y_O, z_O)$ et de rayon R . Le rayon a pour origine $S = (x_S, y_S, z_S)$ et est dirigé par le vecteur unitaire $\vec{v} = (x_v, y_v, z_v)$. Le rayon est paramétré par $P = S + t \vec{v}$ pour t parcourant \mathbb{R} .

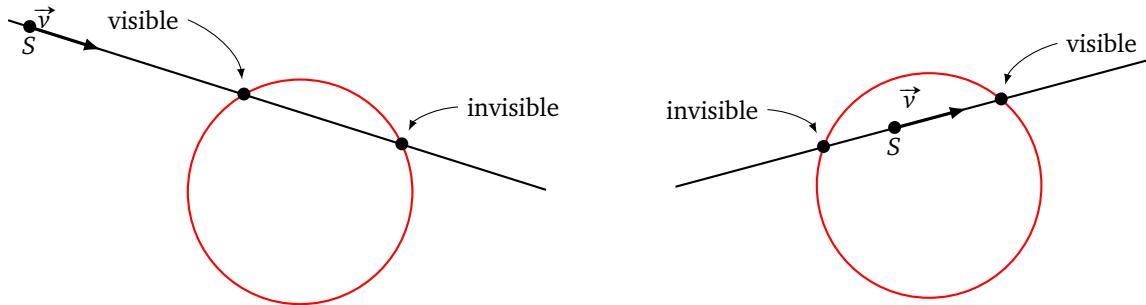
Proposition 7.

Soient

$$\begin{aligned} B &= \vec{v} \cdot \overrightarrow{OS} = x_v(x_S - x_O) + y_v(y_S - y_O) + z_v(z_S - z_O), \\ C &= OS^2 - R^2 = (x_S - x_O)^2 + (y_S - y_O)^2 + (z_S - z_O)^2 - R^2, \\ \Delta &= B^2 - C. \end{aligned}$$

- Si $\Delta < 0$, le rayon n'intersecte pas la sphère.
- Si $\Delta = 0$, le rayon est tangent à la sphère au point P_0 de paramètre $t_0 = -B$.
- Si $\Delta > 0$, le rayon coupe la sphère en deux points P_1 et P_2 de paramètres $t_1 = -B - \sqrt{\Delta}$ et $t_2 = -B + \sqrt{\Delta}$.

Discussion complémentaire : un point P sur la sphère est « devant » le rayon si son paramètre t est positif. Si deux points sont devant le rayon, c'est celui qui est à le plus petit paramètre qui est visible.



Remarque : on peut décider si le rayon coupe (ou pas) la sphère juste par des additions et multiplications de nombres réels. Par contre le calcul des coordonnées des points d'intersection nécessite d'extraire une racine carrée, ce qui est un calcul un peu plus compliqué. En plus nous avons supposé $\|\vec{v}\| = 1$, si ce n'est pas le cas il y a des racines carrées et des divisions à effectuer.

Démonstration. L'équation de la sphère centrée en O et de rayon R est :

$$(x - x_O)^2 + (y - y_O)^2 + (z - z_O)^2 = R^2.$$

Le paramétrage du point $P = (x, y, z)$ du rayon donne :

$$\begin{pmatrix} x \\ y \\ z \end{pmatrix} = \begin{pmatrix} x_S \\ y_S \\ z_S \end{pmatrix} + t \begin{pmatrix} x_v \\ y_v \\ z_v \end{pmatrix}.$$

On reporte ces coordonnées dans l'équation de la sphère :

$$(x_S + tx_v - x_O)^2 + (y_S + ty_v - y_O)^2 + (z_S + tz_v - z_O)^2 = R^2.$$

On développe cette expression pour obtenir une équation polynomiale de degré 2 en t :

$$\begin{aligned} & (x_v^2 + y_v^2 + z_v^2)t^2 \\ & + 2t(x_v(x_S - x_O) + y_v(y_S - y_O) + z_v(z_S - z_O)) \\ & + (x_S - x_O)^2 + (y_S - y_O)^2 + (z_S - z_O)^2 - R^2 = 0 \end{aligned}$$

Comme $\|\vec{v}\|$ est un vecteur supposé unitaire, alors $x_v^2 + y_v^2 + z_v^2 = 1$, l'équation obtenue est donc simplement :

$$t^2 + 2Bt + C = 0.$$

Cette équation de degré 2 en t a pour discriminant réduit $\Delta = B^2 - C^2$. Si $\Delta < 0$ cette équation n'a pas de solution ; si $\Delta = 0$ il y a une solution (double) $t_0 = -B$; si $\Delta > 0$ il y a deux solutions $t_1 = -B - \sqrt{\Delta}$ et $t_2 = -B + \sqrt{\Delta}$. \square

2.3. Sphère : formule géométrique

De nouveau, on suppose \vec{v} de norme 1.

Proposition 8.

1. Soit H le projeté orthogonal de O sur le rayon paramétré par $S + t\vec{v}$. Le paramètre du point H sur le rayon est :

$$t_H = \overrightarrow{SO} \cdot \vec{v}$$

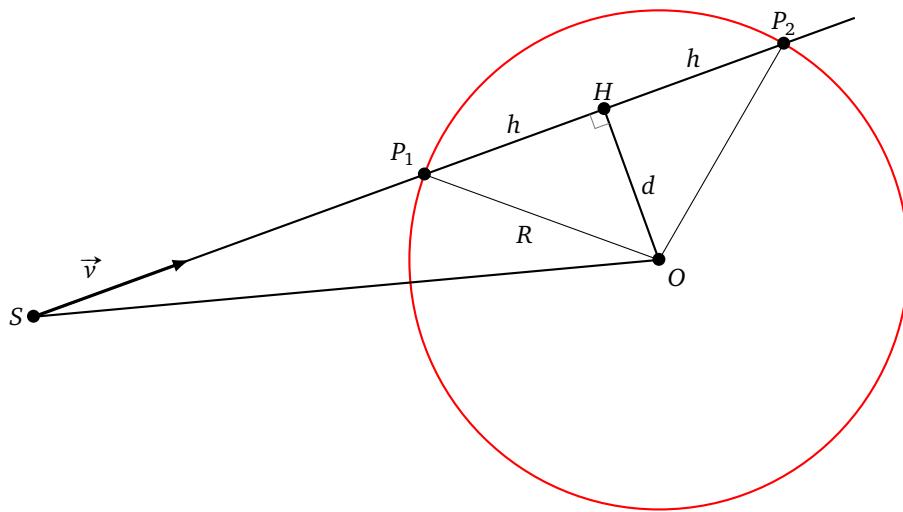
2. Notons $d = OH$. Alors :

$$d^2 = SO^2 - (\overrightarrow{SO} \cdot \vec{v})^2$$

3. Le rayon intersecte la sphère si et seulement si $R^2 - d^2 \geq 0$.

4. Dans ce cas les points d'intersections ont pour paramètres :

$$t_1 = t_H - \sqrt{R^2 - d^2} \quad \text{et} \quad t_2 = t_H + \sqrt{R^2 - d^2}$$



Démonstration.

1. Comme \vec{v} est unitaire alors $t_H = SH$. Or :

$$\overrightarrow{SO} \cdot \vec{v} = (\overrightarrow{SH} + \overrightarrow{HO}) \cdot \vec{v} = \overrightarrow{SH} \cdot \vec{v} = SH.$$

Nous avons utilisé que \overrightarrow{HO} et \vec{v} sont orthogonaux donc leur produit scalaire est nul et aussi que \overrightarrow{SH} et \vec{v} sont colinéaires et de même sens, donc leur produit scalaire est $SH \times \|\vec{v}\|$. Ainsi $t_H = SH = \overrightarrow{SO} \cdot \vec{v}$.

2. Le théorème de Pythagore appliqué dans le triangle SOH (rectangle en H) donne :

$$SO^2 = SH^2 + OH^2,$$

on a noté $d = OH$ et déjà calculé SH , d'où :

$$d^2 = OH^2 = SO^2 - (\overrightarrow{SO} \cdot \vec{v})^2.$$

3. $d = OH$ mesure la distance minimale entre un point du rayon et le centre O de la sphère. Si $d > R$ le rayon ne coupe pas la sphère, si $d = R$ le rayon coupe la sphère tangentiellement en H , si $d < R$ le rayon coupe la sphère en deux points distincts.
4. Dans le cas $d \leq R$, le rayon coupe la sphère en deux points P_1 et P_2 symétriques l'un de l'autre par rapport à H . Considérons le triangle OP_1H (rectangle en H), le théorème de Pythagore donne :

$$OP_1^2 = OH^2 + HP_1^2.$$

On a $R = OP_1$, $d = OH$ et on note $h = HP_1$, ainsi $h^2 = R^2 - d^2$. Donc $h = \sqrt{R^2 - d^2}$. Comme \vec{v} est unitaire alors $t_1 = t_H - h$, et comme P_2 est le symétrique de P_1 par rapport à H , alors $t_2 = t_H + h$.

□

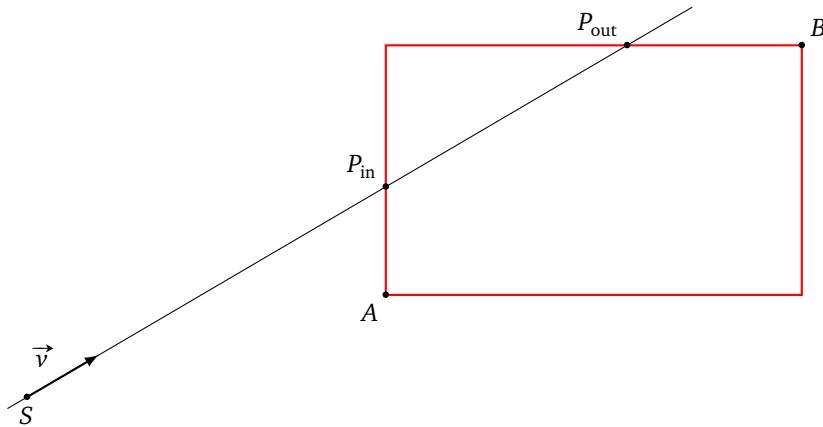
Exercice. Reprendre les calculs pour un vecteur \vec{v} non unitaire. Vérifier que les trois premiers points utilisent seulement $\|\vec{v}\|^2$ (et donc il n'y a pas besoin d'extraire une racine carrée pour obtenir $\|\vec{v}\|$).

3. Intersection avec une boîte

Pour décider si un objet complexe va être intersecté par un rayon on peut commencer par inclure l'objet dans une boîte (*bounding box*). Si le rayon ne coupe pas la boîte (nous allons voir que c'est un calcul facile) alors nous sommes sûrs que le rayon ne coupe pas l'objet et on s'épargne alors des calculs compliqués. Si le rayon coupe la boîte, alors on peut lancer le calcul pour savoir si le rayon coupe notre objet et, si oui, en quels points.

3.1. Explications dans le plan

Dans le plan une **boîte** est un rectangle dont les côtés sont parallèles aux axes. Nous définissons une boîte par son coin inférieur gauche $A(x_A, y_A)$ et son coin supérieur droit $B(x_B, y_B)$.

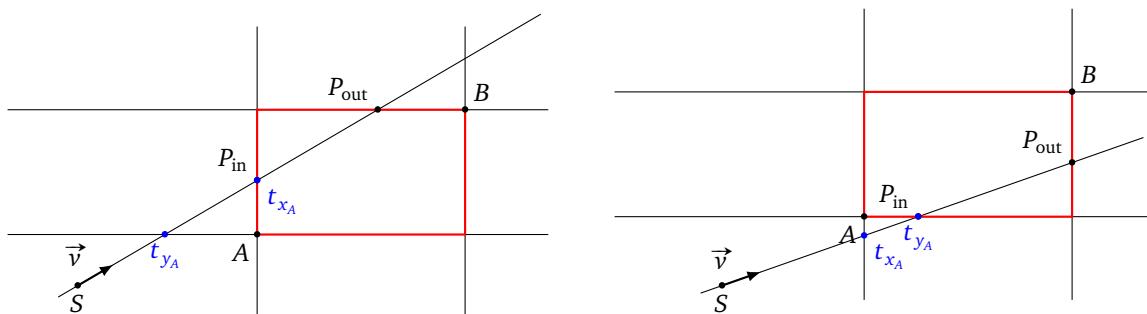


Nous lançons un rayon issu de $S(x_S, y_S)$ dirigé par le vecteur $\vec{v} = (x_v, y_v)$. On veut savoir si le rayon coupe la boîte, et si oui, calculer les valeurs t_{in} et t_{out} des points $P = S + t \vec{v}$ d'entrée et de sortie. Pour simplifier les explications nous supposons que le rayon pointe vers la droite et vers le haut ($x_v > 0$ et $y_v > 0$). Le rayon coupe la droite verticale d'équation ($x = x_A$) (qui supporte le côté gauche de la boîte) pour le paramètre t_{x_A} qui vérifie l'équation $x_S + t x_v = x_A$, donc

$$t_{x_A} = \frac{x_A - x_S}{x_v}.$$

De même le rayon coupe la droite horizontale d'équation ($y = y_A$) (qui supporte le bas de la boîte) pour le paramètre :

$$t_{y_A} = \frac{y_A - y_S}{y_v}.$$

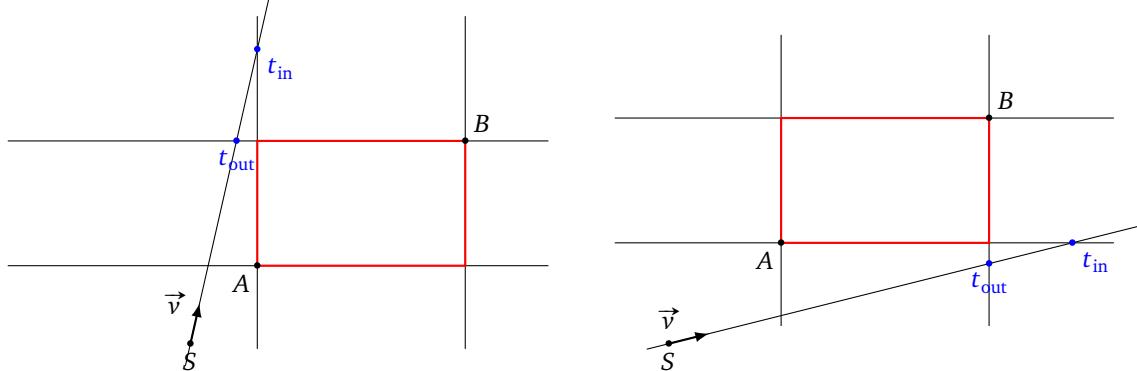


On définit $t_{\text{in}} = \max(t_{x_A}, t_{y_A})$. Si le rayon coupe la boîte, alors le point d'entrée est P_{in} , de paramètre t_{in} . De même pour le point B , on pose

$$t_{x_B} = \frac{x_B - x_S}{x_v} \quad t_{y_B} = \frac{y_B - y_S}{y_v} \quad t_{\text{out}} = \min(t_{x_B}, t_{y_B}).$$

Si le rayon coupe la boîte, alors le point de sortie est P_{out} , de paramètre t_{out} .

Que se passe-t-il si le rayon n'intersecte pas la boîte ? Dans ce cas les valeurs t_{in} et t_{out} sont bien définies mais on a $t_{\text{in}} > t_{\text{out}}$ (le point d'entrée serait après le point de sortie !).



Résumons nos observations :

Proposition 9.

Considérons le cas $x_v > 0$ et $y_v > 0$. Soient :

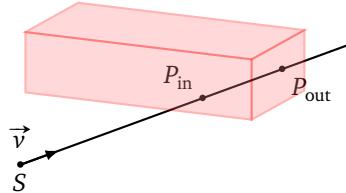
$$t_{\text{in}} = \max\left(\frac{x_A - x_S}{x_v}, \frac{y_A - y_S}{y_v}\right) \quad \text{et} \quad t_{\text{out}} = \min\left(\frac{x_B - x_S}{x_v}, \frac{y_B - y_S}{y_v}\right).$$

Le rayon intersecte la boîte si et seulement si $t_{\text{in}} \leq t_{\text{out}}$. Dans ce cas le point d'entrée P_{in} est de paramètre t_{in} et celui de sortie P_{out} est de paramètre t_{out} .

Noter que la source S peut être à l'intérieur de la boîte, dans ce cas le rayon intersecte toujours la boîte mais le point P_{in} est avant la source et donc le premier point qui atteint la boîte après la source est P_{out} .

3.2. Algorithme dans l'espace

Une **boîte** de l'espace est un parallélépipède rectangle dont les faces sont parallèles aux axes. Elle est définie par deux sommets opposés $A(x_A, y_A, z_A)$ et $B(x_B, y_B, z_B)$. Le rayon est issu de $S(x_S, y_S, z_S)$ et dirigé par un vecteur non nul quelconque $\vec{v}(x_v, y_v, z_v)$.



Voici le calcul du point d'entrée P_{in} , de paramètre t_{in} , et celui de sortie P_{out} , de paramètre t_{out} , s'ils existent.

Algorithme.

- $t_{\text{in}} := -\infty$, $t_{\text{out}} := +\infty$ (ou bien une très petite et une très grande valeur).
- Si $x_v \neq 0$, poser :

$$t_{x_A} := \frac{x_A - x_S}{x_v} \quad t_{x_B} := \frac{x_B - x_S}{x_v}$$

et

$$t_{\text{in}} := \max(t_{\text{in}}, \min(t_{x_A}, t_{x_B})) \quad t_{\text{out}} := \min(t_{\text{out}}, \max(t_{x_A}, t_{x_B})).$$

- Si $y_v \neq 0$, poser :

$$t_{y_A} := \frac{y_A - y_S}{y_v} \quad t_{y_B} := \frac{y_B - y_S}{y_v}$$

et

$$t_{\text{in}} := \max(t_{\text{in}}, \min(t_{y_A}, t_{y_B})) \quad t_{\text{out}} := \min(t_{\text{out}}, \max(t_{y_A}, t_{y_B})).$$

- Si $z_v \neq 0$, poser :

$$t_{z_A} := \frac{z_A - z_S}{z_v} \quad t_{z_B} := \frac{z_B - z_S}{z_v}$$

et

$$t_{\text{in}} := \max(t_{z_A}, \min(t_{z_B}, t_{z_B})) \quad t_{\text{out}} := \min(t_{\text{out}}, \max(t_{z_A}, t_{z_B})).$$

- Si $t_{\text{in}} \leq t_{\text{out}}$ alors le rayon intersecte la boîte en P_{in} (de paramètre t_{in}) et P_{out} (de paramètre t_{out}), sinon le rayon ne coupe pas la boîte.

Note : pour accélérer l'affichage, il est important de pouvoir effectuer les calculs en parallèle. Une instruction conditionnelle (comme si ... alors ..., sinon ...) peut poser des problèmes. C'est la situation rencontrée lorsque qu'on considère la définition classique du maximum :

$$\max(a, b) = \begin{cases} a & \text{si } a \geq b, \\ b & \text{sinon.} \end{cases}$$

Cependant il est possible de contourner ce problème et de définir le maximum par une seule formule :

$$\max(a, b) = \frac{a + b + |a - b|}{2}.$$

Note. La section sur les polygones est inspirée de Paul Bourke : [Determining if a point lies on the interior of a polygon](#).

Lumière

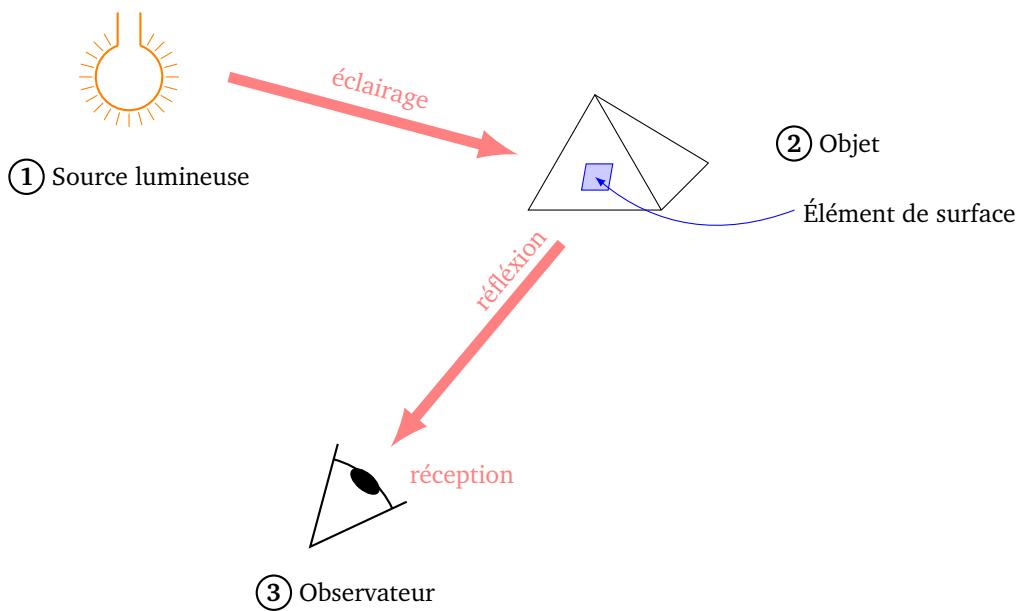
Comment une scène est-elle éclairée ? Quelle est la couleur des objets illuminés ?

1. Physique de la lumière

1.1. Vue d'ensemble

La lumière possède à la fois les propriétés d'une onde et des corpuscules (photons). La lumière et l'éclairage sont des phénomènes complexes, difficiles à modéliser. Ici nous allons faire des hypothèses simplificatrices :

- L'éclairage est direct, avec une seule source lumineuse. Si on avait plusieurs sources, il suffirait de faire les calculs pour chaque source. Par contre on considère que les objets sont uniquement éclairés par les sources lumineuses et non par réflexion de la lumière sur d'autres objets (comme le serait l'éclairage via un miroir par exemple).
- Il n'y a pas de transparence. Un objet avec transparence réfléchit une partie de la lumière reçue et laisse passer une autre partie. Nos objets seront opaques.



Le but est de déterminer la couleur d'un objet perçu par un observateur (œil, caméra, capteur...). Dans la pratique, on décompose l'objet en surfaces élémentaires (petits carrés ou petits triangles par exemple) et on détermine la couleur de chaque surface élémentaire.

Cette couleur dépend de la *source lumineuse* (position, direction, intensité, couleur...), de l'*objet* observé (position, orientation, couleur, texture...) ainsi que de la position de l'*observateur*.

1.2. Physique

Le vocabulaire permet de distinguer ce qui est émis par la source lumineuse de ce qui reçu/perçu par l'objet/l'observateur.

- **Luminosité**/Intensité lumineuse/**Éclairage**/*Luminance*.

L'unité désuète d'intensité lumineuse est le *watt* (exemple une ampoule classique de 75 W, un projecteur de 500 W). Cela correspondait à la puissance consommée par l'éclairage avant l'apparition des ampoules LED. L'unité officielle est le *lumen* (lm) : elle compte le flux d'énergie lumineuse (puissance) pour un angle d'un stéradian. Exemple : l'éclairage d'une pièce de 10 ou 15 m² nécessite une intensité totale d'environ 3000 à 5000 lm.

La luminosité c'est donc ce qui arrive vers l'objet, ce n'est pas ce qui est réfléchi et observé.

- **Illuminate****nation**/Éclairement/ *Illuminance*.

L'illumination c'est la façon dont la lumière arrive vers l'objet. Elle dépend des propriétés physiques de l'objet (position, orientation...). L'unité d'éclairement est le *lux* (lx) : c'est le flux d'énergie reçue par une unité de surface. Exemple : 1 lumen illuminant une surface de 1 m² correspond à 1 lux.

1.3. Couleur

La lumière possède sa couleur propre, l'objet possède aussi sa couleur propre. Le but est d'attribuer une couleur perçue à (une surface élémentaire de) l'objet.

- Une lumière monochromatique est caractérisée par une seule longueur d'onde : de 400 nm (violet) à 800 nm (rouge) pour les longueurs d'ondes visibles. C'est l'analogie de jouer une seule note de musique sur un piano.
- La lumière est une superposition d'ondes monochromatiques (comme un accord de musique où plusieurs touches sont jouées en même temps).
- L'objet absorbe certaines ondes (de longueurs d'onde déterminées) et en réfléchit d'autres. Par exemple, si un objet, éclairé par une lumière blanche, absorbe le bleu et réfléchit le rouge et le vert alors sa couleur perçue sera jaune (= rouge + vert).

Nous caractériserons une couleur via le système RGB (*Red/Green/Blue*).

- Une couleur est caractérisée par trois réels (r, g, b) avec $r, g, b \in [0, 1]$. Exemples : $(1, 0, 0)$ est la couleur rouge, $(1, 1, 0)$ est du jaune...
- **Addition.** L'addition de deux couleurs se fait terme à terme, sans dépasser 1. C'est par exemple l'opération naturelle lorsque on considère l'éclairage total produit par deux sources lumineuses. Si $Cl_1 = (r_1, g_1, b_1)$ et $Cl_2 = (r_2, g_2, b_2)$ alors l'addition totale est $(r_1 + r_2, g_1 + g_2, b_1 + b_2)$. Cependant il faut prendre garde à ce qu'aucune composante ne dépasse 1. Ainsi l'addition de couleurs est définie par la formule :

$$Cl_1 \oplus_{cl} Cl_2 = (\min(r_1 + r_2, 1), \min(g_1 + g_2, 1), \min(b_1 + b_2, 1))$$

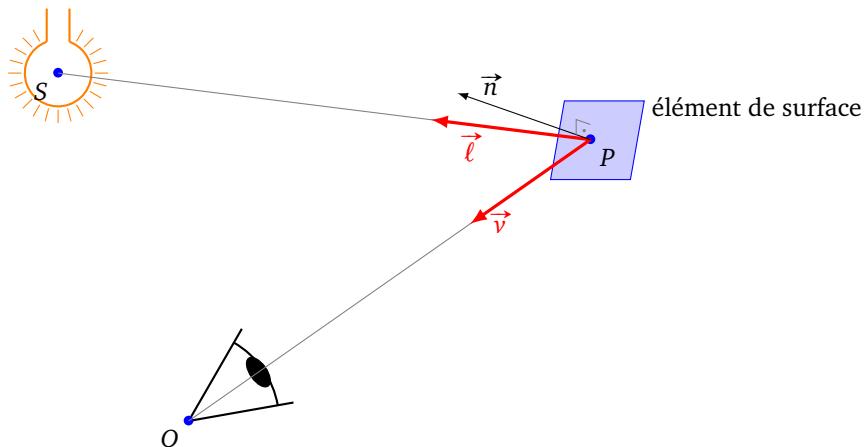
- **Multiplication.** La multiplication s'effectue aussi terme à terme :

$$Cl_1 \otimes_{cl} Cl_2 = (r_1 r_2, g_1 g_2, b_1 b_2)$$

Lorsque une lumière de couleur Cl_1 éclaire un objet de couleur Cl_2 , la multiplication des couleurs renvoie la couleur perçue. Exemple : si une lumière de couleur $Cl_1 = (1, 0, 1)$ (violet) éclaire un objet de couleur $Cl_2 = (1, 1, 0)$ (jaune) alors la multiplication des couleurs donne $Cl_1 \otimes_{cl} Cl_2 = (1, 0, 0)$, ainsi la couleur perçue de l'objet est rouge. Pour la multiplication de couleurs sombres, on peut ajuster la formule par un facteur multiplicatif.

1.4. Notations

Introduisons les notations utilisées dans la suite du chapitre.



Trois points.

- S : le point origine de la source lumineuse,
- P : le point centré sur la surface élémentaire de l'objet,
- O : le point où on place la caméra/l'œil/l'observateur.

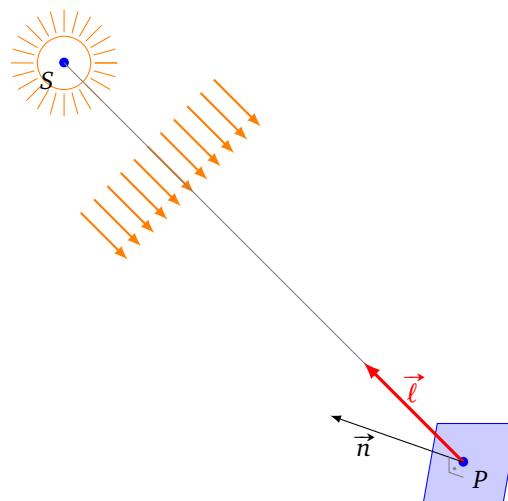
Trois vecteurs.

- $\vec{\ell}$: le vecteur unitaire issu de P dirigé vers la source lumineuse,
- \vec{n} : le vecteur unitaire orthogonal à la surface élémentaire,
- \vec{v} : le vecteur unitaire issu de P et dirigé vers l'observateur O .

2. Éclairage

2.1. Lumière directionnelle

Les rayons lumineux arrivent tous avec la même direction. C'est le cas par exemple de la lumière du soleil, ou d'une source lumineuse très éloignée.



Cette lumière est caractérisée par :

- une direction : si \vec{u} est le vecteur qui indique la direction des rayons, alors

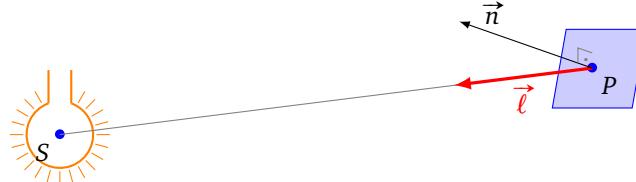
$$\vec{\ell} = -\frac{\vec{u}}{\|\vec{u}\|}.$$

- une intensité : i_ℓ .

L'intensité i_ℓ et cette direction $\vec{\ell}$ sont constantes, quelle que soit la position du point P .

2.2. Lumière ponctuelle

La lumière « rayonne » dans toutes les directions depuis la source, comme par exemple une ampoule.



Le vecteur $\vec{\ell}$ dépend de la position du point P :

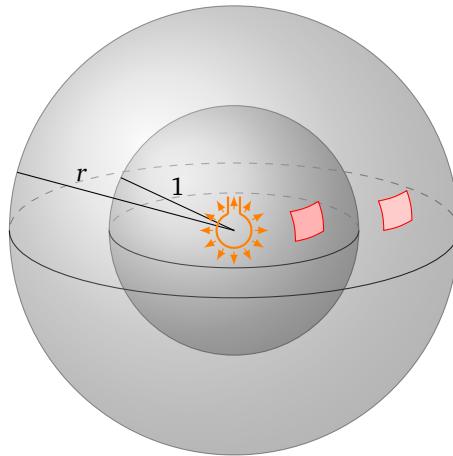
$$\vec{\ell} = \frac{\overrightarrow{PS}}{\|\overrightarrow{PS}\|}.$$

L'énergie lumineuse i_ℓ reçue au point P diminue lorsque P s'éloigne de la source S . Selon quelle loi varie cette intensité ? Notons $r = PS$ la distance entre le point P et la source. Notons i_0 l'intensité reçue à une distance d'une unité (par exemple 1 m).

Proposition 1.

$$i_\ell = \frac{i_0}{r^2}$$

Justification. Notons E_0 l'énergie lumineuse qui traverse une sphère de rayon 1 centrée sur la source S pendant une unité de temps (ex. 1 s). Cette énergie se répartit uniformément sur une surface d'aire 4π (rappel : l'aire d'une sphère de rayon r est $4\pi r^2$).



Un tout petit peu plus tard ces rayons traversent la sphère de rayon r . L'énergie E_0 est conservée mais cette fois se répartit sur une surface d'aire $4\pi r^2$. Ainsi chaque élément de surface reçoit beaucoup moins d'énergie. Par exemple, 1 m^2 de la sphère de rayon 1 reçoit une énergie $\frac{E_0}{4\pi}$, alors que 1 m^2 de la sphère de rayon r reçoit une énergie $\frac{E_0}{4\pi r^2}$.

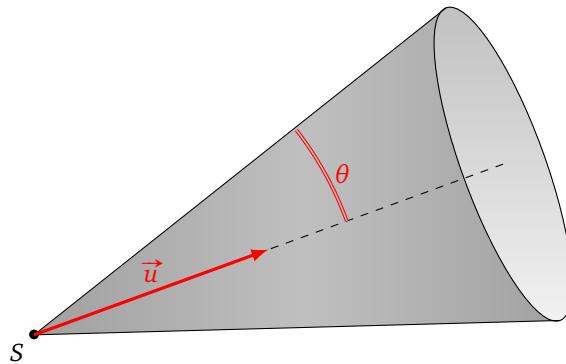
Variante avec plus de paramètres.

$$i_\ell = \frac{1}{\alpha r^2 + \beta r + \gamma}$$

Les paramètres α, β, γ permettent davantage de réglages pour obtenir un éclairage plus réaliste. Si $\alpha = \frac{1}{i_0}$, $\beta = 0$, $\gamma = 0$, on retrouve la loi précédente.

2.3. Spot

Pour un spot, les rayons lumineux rayonnent à partir de la source, mais sont limités par un cône. Par exemple : un spot, une lampe de poche, un projecteur.



Le cône est caractérisé par un axe central \vec{u} (supposé unitaire) et un demi-angle θ .

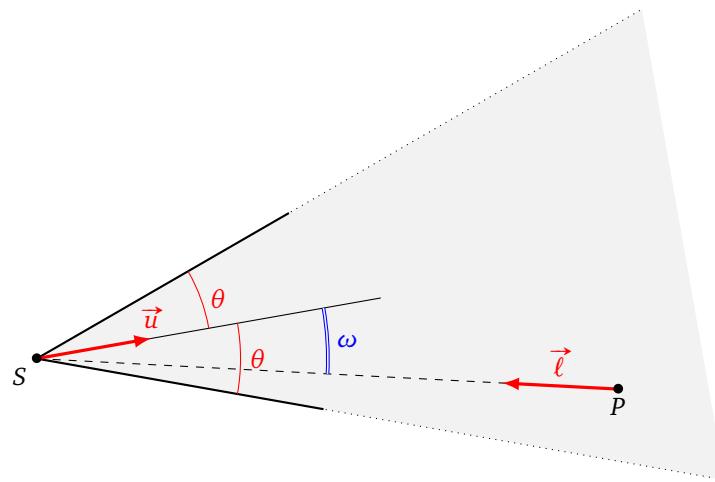
Proposition 2.

Le point P appartient au cône de sommet S , d'axe \vec{u} et de demi-angle θ si et seulement si :

$$\cos(\omega) \geq \cos(\theta)$$

où l'angle ω est l'angle de P par rapport à l'axe du cône :

$$\cos(\omega) = -\vec{\ell} \cdot \vec{u}$$



Démonstration. Tout d'abord on obtient l'angle ω entre \vec{SP} et \vec{u} par la relation $\vec{\ell} \cdot \vec{u} = -\cos(\omega)$, donc :

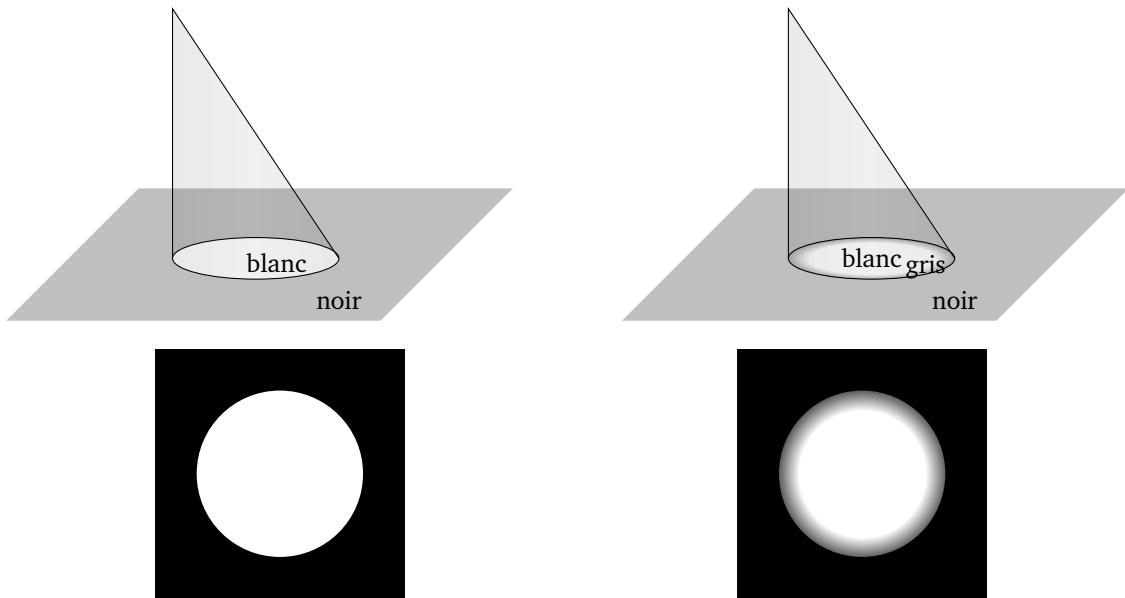
$$\cos(\omega) = -\vec{\ell} \cdot \vec{u}.$$

De plus P appartient au cône si l'angle ω est plus petit que l'angle θ , plus précisément lorsque $|\omega| \leq \theta$, ce qui équivaut à $\cos(\omega) \geq \cos(\theta)$. D'où le résultat. \square

L'intensité lumineuse reçue par un point du cône suit la même loi que celle de la lumière ponctuelle :

$$i_\ell = \begin{cases} \frac{i_0}{r^2} & \text{si } \cos(\omega) \geq \cos(\theta) \\ 0 & \text{sinon} \end{cases}$$

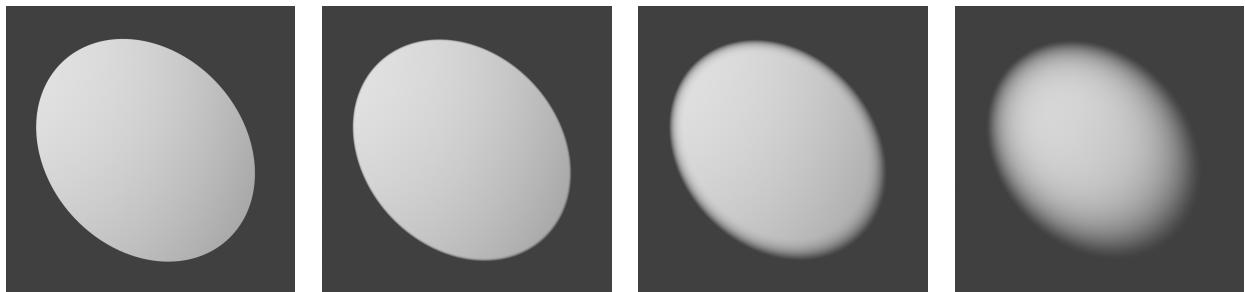
On rappelle que $r = SP$.



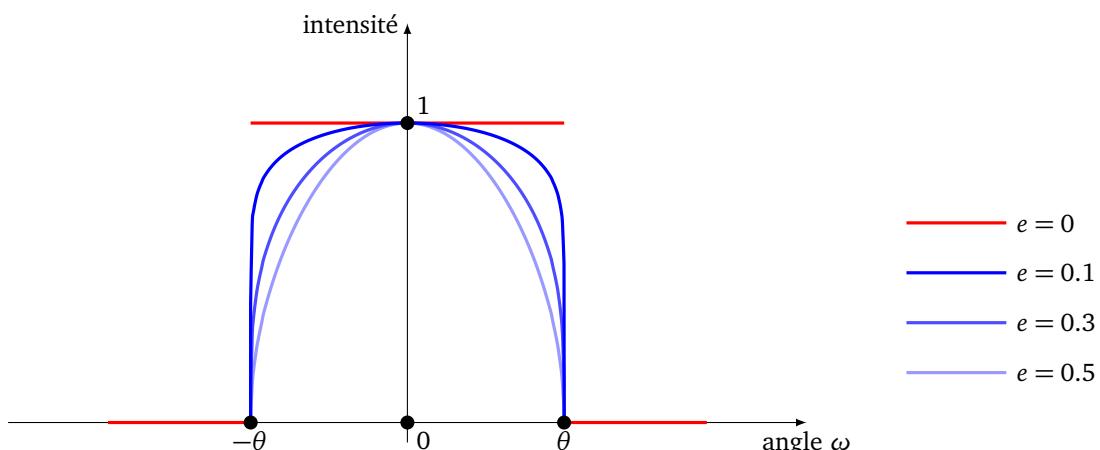
Pour atténuer le passage zone obscure/zone éclairée, on peut effectuer une transition douce au bord du cône. On assombrit progressivement la zone éclairée située à angle ω lorsque celui-ci se rapproche de l'angle θ , selon la formule :

$$i_\ell = \begin{cases} \frac{i_0}{r^2} (\cos(\omega))^{e_{\text{att}}} & \text{si } \cos(\omega) \geq \cos(\theta) \\ 0 & \text{sinon} \end{cases}$$

L'exposant d'atténuation e_{att} permet de paramétrer cette transition, qui est plus condensée pour des petites valeurs de l'exposant.



Voici les allures de $(\cos(\omega))^{e_{\text{att}}}$ pour différentes valeurs de l'exposant. Pour $e_{\text{att}} = 0$ le passage zone éclairée/zone non-éclairée se fait sans transition. Pour $e_{\text{att}} > 0$, il y a plateau autour du centre du cône (pour ω proche de 0) et une atténuation plus ou moins rapide sur les bords du cône (pour $|\omega|$ proche de θ).



Évidemment on peut considérer une version plus générale avec :

$$i_\ell = \frac{1}{\alpha r^2 + \beta r + \gamma} (\cos(\omega))^{e_{\text{att}}}.$$

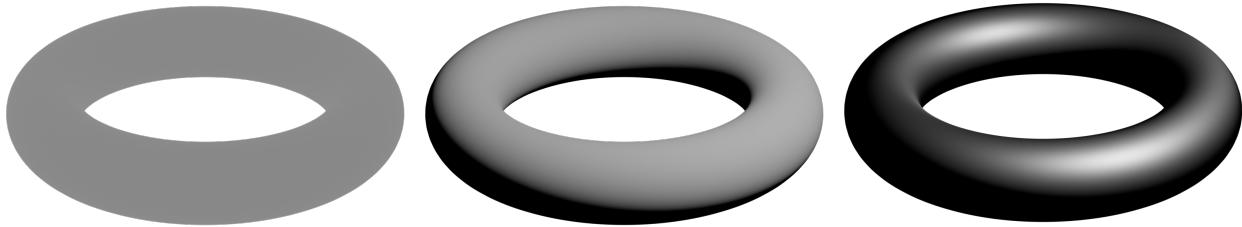
3. Illumination

3.1. Modèle de Phong

La couleur dont est perçu un objet dépend de son éclairage (intensité, direction, couleur...) mais aussi de l'objet lui-même (couleur, matière, brillant/mat...).

Nous modélisons la couleur perçue via le **modèle de Phong**. Un calcul est à effectuer pour chaque élément de surface. La couleur s'obtient comme la superposition de quatre types de luminosité dont on additionne les couleurs :

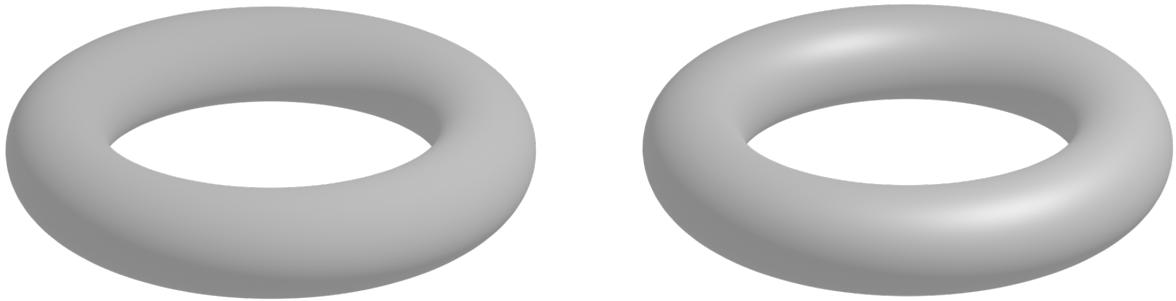
- luminosité irradiante Cl_{irr} (facultative),
- luminosité ambiante Cl_{amb} ,
- luminosité diffuse Cl_{diff} ,
- luminosité spéculaire Cl_{spec} .



Lumière ambiante

Lumière diffuse

Lumière spéculaire



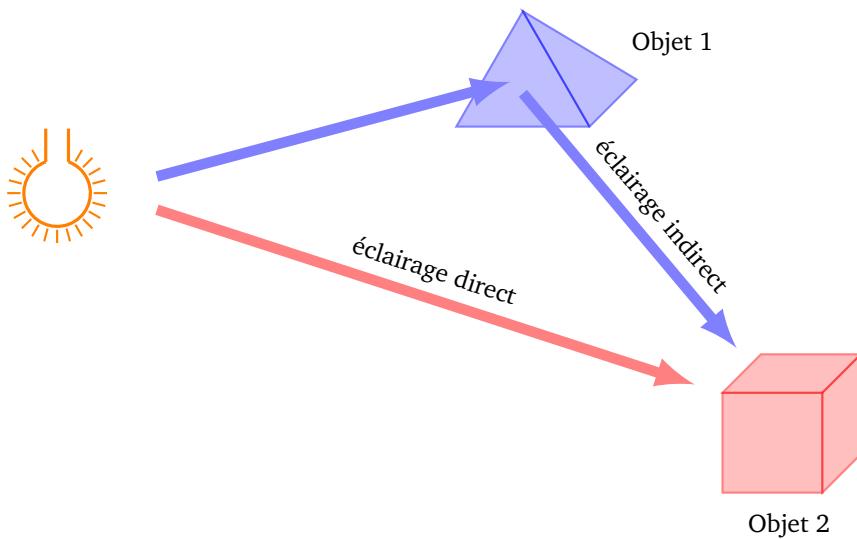
Lumière ambiante et diffuse

Lumière diffuse, ambiante et spéculaire

La couleur perçue est l'addition de ces couleurs :

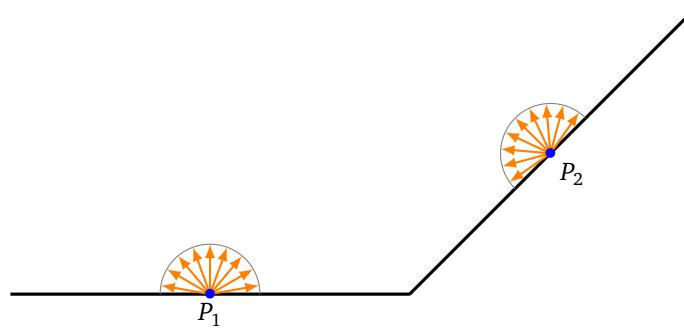
$$Cl_{\text{perçue}} = Cl_{\text{irr}} \oplus_{cl} Cl_{\text{amb}} \oplus_{cl} Cl_{\text{diff}} \oplus_{cl} Cl_{\text{spec}}$$

Ce modèle est empirique, et n'est pas fondé sur une réalité physique, mais a fait ses preuves. Il est utilisé dans les moteurs 3D en attendant le *ray-tracing* généralisé. Nous allons essentiellement considérer une seule source lumineuse (pour plusieurs sources on effectue les calculs pour chaque source et on additionne les résultats). Nous omettons ici l'éclairage indirect issu de la réflexion de la lumière sur un autre objet.



3.2. Luminosité irradiante

La **luminosité irradiante** est une lumière émise par l'objet lui-même, même en l'absence d'éclairage, comme une sorte de rayonnement de l'objet dans toutes les directions.



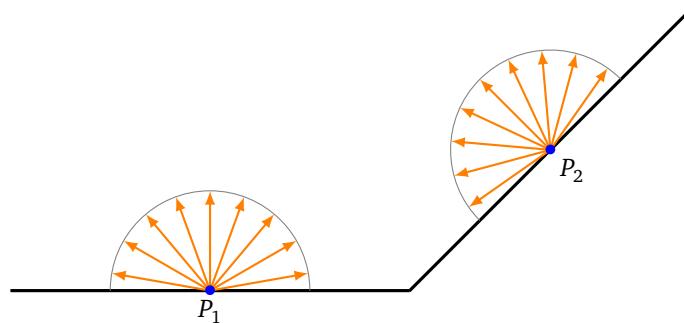
La couleur associée est notée :

$$Cl_{\text{irr}}$$

Cette couleur RGB est la même pour tout point de l'objet, quelle que soit la position de l'observateur. Cette luminosité est de faible intensité et n'est pas considérée comme une source lumineuse pour les autres objets.

3.3. Luminosité ambiante

La **luminosité ambiante** correspond à une lumière qui n'a pas de source précise et éclaire partout, et dans toutes les directions, de la même façon. C'est par exemple le cas d'une scène en extérieur par ciel nuageux, ou bien dans une pièce avec plusieurs fenêtres ayant un voilage.



La couleur perçue Cl_{amb} dépend de la couleur Cl_{source} et de l'intensité i_{source} de l'éclairage, mais aussi de la couleur propre de l'objet Cl_{objet} :

$$Cl_{\text{amb}} = i_{\text{source}} Cl_{\text{source}} \otimes_{cl} Cl_{\text{objet}}$$

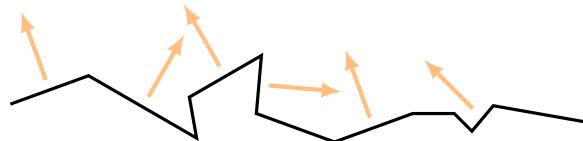
Remarques :

- On rappelle que \otimes_{cl} désigne la multiplication terme à terme des composantes r, g, b .
- Cette lumière éclaire tous les objets et toutes leurs faces.
- C'est un éclairage dont l'intensité reste modérée et qui doit rester discret. Il a pour défaut « d'aplatir » la scène.
- La différence avec la luminosité irradiante est que la luminosité ambiante a une intensité plus forte donc elle pourrait éclairer les autres objets (mais ici nous n'en tiendrons pas compte).

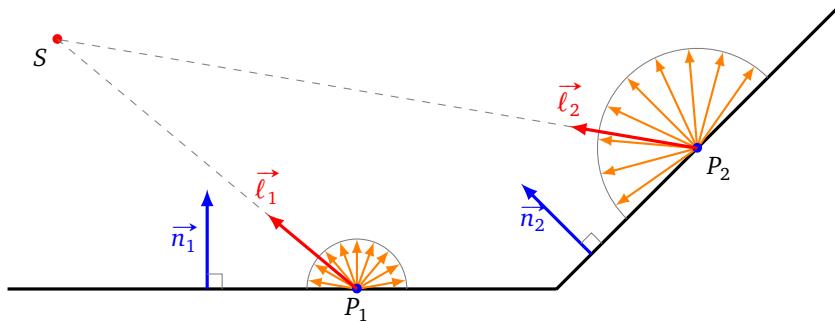
3.4. Luminosité diffuse

La **luminosité diffuse** est une caractéristique associée aux matériaux mats (ex. : bois, cuir...) dont l'aspect microscopique est rugueux et conduit à une réflexion de la lumière dans toutes les directions.

Voici un zoom à un niveau microscopique d'une surface mate :



La lumière reçue se diffuse dans toutes les directions mais l'intensité diffuse dépend de l'intensité de la lumière reçue et donc de la position de l'objet par rapport à l'éclairage.



La formule de la couleur diffuse est résumée en :

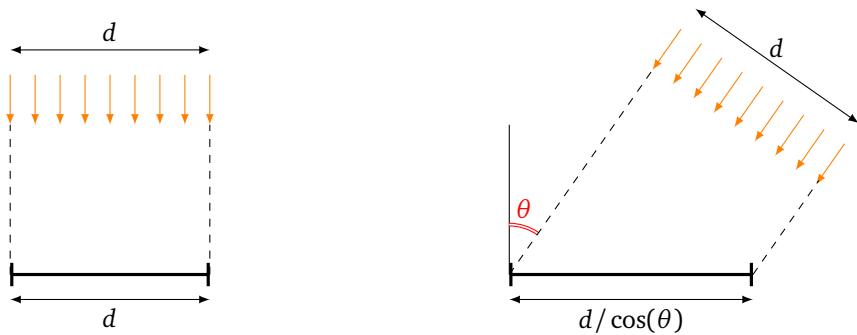
$$Cl_{\text{diff}} = i_{\text{source}} \max(\vec{l} \cdot \vec{n}, 0) Cl_{\text{source}} \otimes_{cl} Cl_{\text{objet}}$$

où i_{source} et Cl_{source} sont l'intensité et la couleur de la source lumineuse et Cl_{objet} est la couleur propre de l'objet.

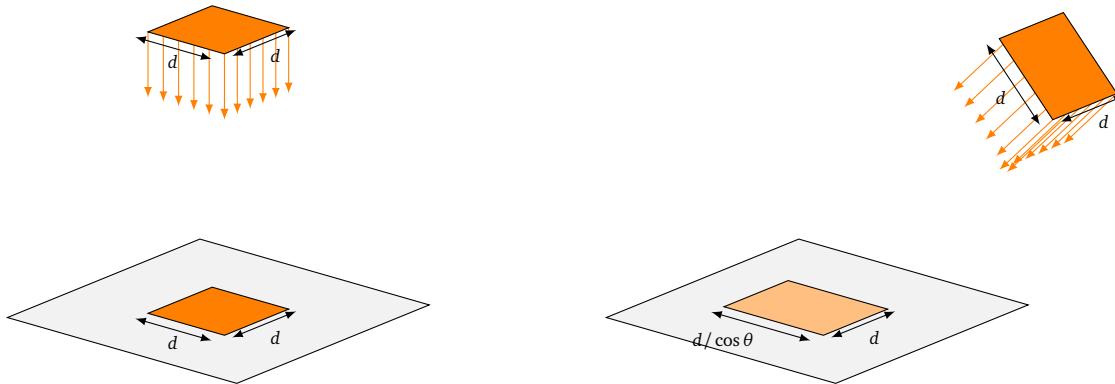
Expliquons le terme $\max(\vec{l} \cdot \vec{n}, 0)$. Tout d'abord l'éclairage est situé derrière l'objet (autrement dit éclaire la face non visible) lorsque $\vec{l} \cdot \vec{n} < 0$, dans ce cas $\max(\vec{l} \cdot \vec{n}, 0) = 0$ et donc la couleur diffuse est noire $Cl_{\text{diff}} = (0, 0, 0)$.

Le produit scalaire $\vec{l} \cdot \vec{n}$ module l'intensité en fonction de l'orientation de la surface éclairage vis à vis de la source. Si la surface est perpendiculaire aux rayons lumineux alors \vec{n} et \vec{l} sont égaux et leur produit scalaire vaut 1. Si la surface n'est pas perpendiculaire, l'intensité est diminuée d'un facteur $\vec{l} \cdot \vec{n} < 1$.

Justification. Considérons l'énergie E_0 d'une bande de rayons lumineux de largeur d qui atteint une surface. Dans un premier cas cette surface est perpendiculaire aux rayons et l'énergie E_0 se répartit sur une largeur d . Dans un second cas la surface est inclinée d'un angle θ , et alors la même énergie E_0 se répartit sur une largeur plus grande, égale à $\frac{d}{\cos(\theta)}$.



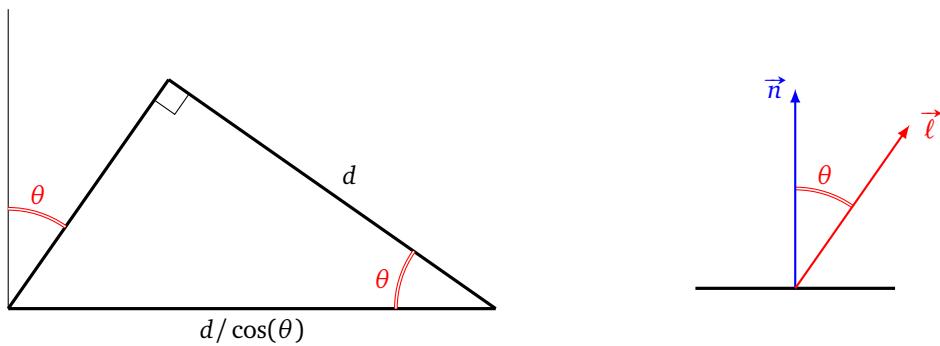
Si maintenant on raisonne en énergie reçue par unité de surface (par exemple 1 m^2) alors chaque unité de surface inclinée reçoit moins d'énergie avec un coefficient multiplicatif de $\cos(\theta)$ (qui est inférieur ou égal à 1).



Remarque. C'est aussi la raison pour laquelle il fait plus froid en hiver qu'en été : en hiver, le Soleil est plus bas dans le ciel et l'énergie reçue s'étale sur une plus grande surface.

Calculs. On note $\theta \in]-\pi, \pi]$ l'angle entre les vecteurs \vec{n} et \vec{l} . On exprime facilement θ par la relation :

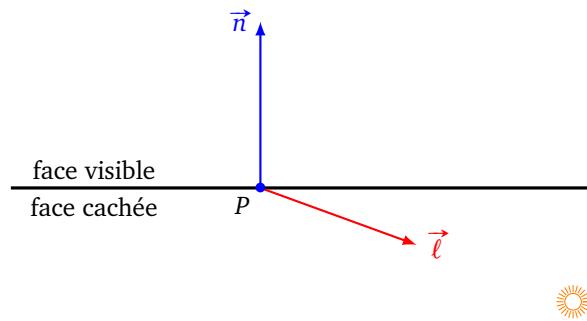
$$\cos(\theta) = \vec{l} \cdot \vec{n}$$



Et les relations trigonométriques usuelles, montrent que la bande de largeur d illumine sur la surface inclinée une bande de largeur $\frac{d}{\cos(\theta)}$. Par conséquence l'intensité lumineuse diffuse s'exprime par :

$$i_{\text{diff}} = i_{\text{source}} \times \cos(\theta) = i_{\text{source}} \vec{l} \cdot \vec{n}.$$

Cela explique presque complètement les termes de la formule. Cependant on ne diffuse pas de lumière si la source est derrière l'objet, c'est-à-dire si $|\theta| > \frac{\pi}{2}$ (dans ce cas la lumière arrive sur la face cachée). Or, pour nos angles, $|\theta| > \frac{\pi}{2}$ équivaut à $\cos(\theta) < 0$, donc si $\cos(\theta) < 0$ on fixe $i_{\text{diff}} = 0$.

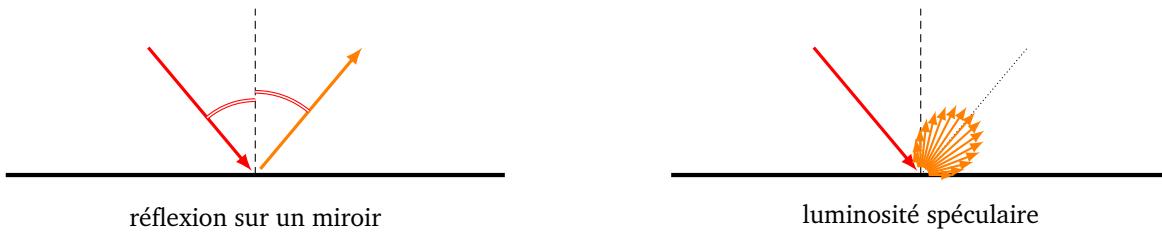


Résumons :

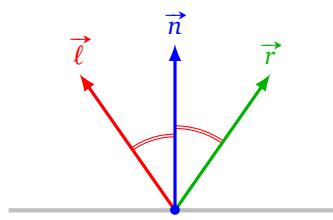
$$i_{\text{diff}} = i_{\text{source}} \max(0, \cos(\theta)) = \begin{cases} i_{\text{source}} \cos(\theta) & \text{si } \cos(\theta) \geq 0, \\ 0 & \text{sinon.} \end{cases}$$

3.5. Luminosité spéculaire

La luminosité diffuse vue précédemment renvoie uniformément la lumière dans toutes les directions. Mais en réalité, la lumière se réfléchit davantage par réflexion sur la surface : c'est la **luminosité spéculaire**. Elle s'observe sur les objets brillants (plastique, métal poli...), le cas extrême étant celui d'un miroir sur lequel un rayon lumineux est renvoyé dans une seule direction. Cependant, si la surface est moins lisse, les réflexions s'étalent autour d'une direction privilégiée (la distribution étant centrée autour d'un axe principal de réflexion).



Attention ! La luminosité spéculaire perçue dépend de la position de l'observateur. Elle demande donc davantage de calculs que les luminosités précédentes. Par contre elle apporte une grosse touche de réalisme. Notons \vec{r} l'axe principal de la réflexion, qui est le vecteur symétrique de \vec{l} (qui pointe vers la source lumineuse S) par rapport à \vec{n} (orthogonal à la surface). Notons \vec{v} le vecteur, issu de P , pointant vers l'observateur O . Nous supposons que $\vec{l}, \vec{n}, \vec{r}, \vec{v}$ sont des vecteurs unitaires.



Lemme 1.

$$\vec{r} = 2(\vec{l} \cdot \vec{n})\vec{n} - \vec{l}$$

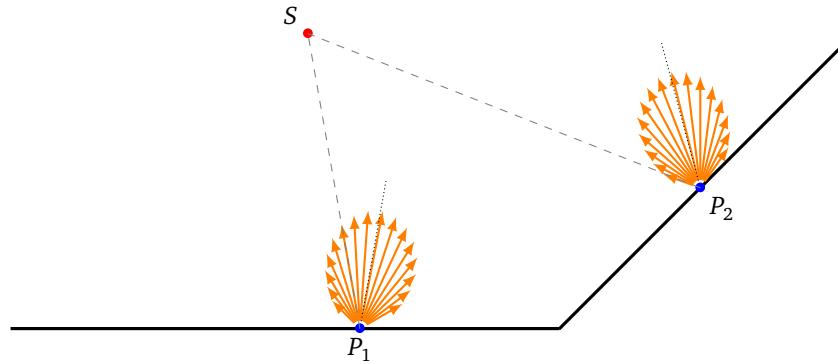
Voici la formule de la couleur de la luminosité spéculaire.

Si $\vec{l} \cdot \vec{n} \geq 0$ et $\vec{r} \cdot \vec{v} \geq 0$ alors :

$$Cl_{\text{spec}} = i_{\text{source}} (\vec{r} \cdot \vec{v})^{e_{\text{spec}}} Cl_{\text{source}} \otimes_{cl} Cl_{\text{objet}}$$

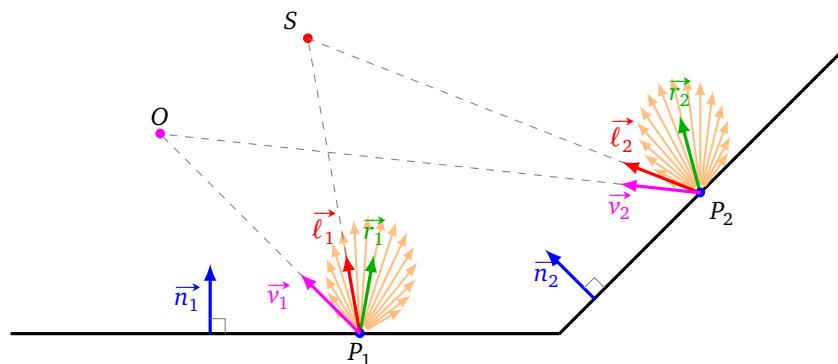
Si $\vec{\ell} \cdot \vec{n} < 0$ ou $\vec{r} \cdot \vec{v} < 0$ alors, la surface n'est pas visible ou pas éclairée et on pose :

$$Cl_{\text{spec}} = (0, 0, 0)$$



La formule dépend :

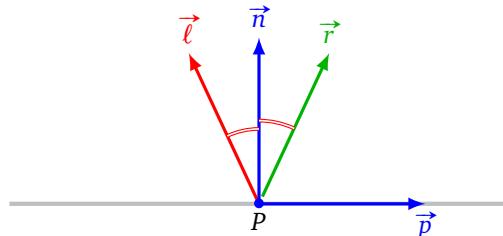
- de i_{source} et Cl_{source} : l'intensité et la couleur de la source lumineuse,
- de la position de l'axe principal de réflexion \vec{r} par rapport à la direction \vec{v} vers l'observateur,
- d'un exposant e_{spec} qui est un paramètre d'étalement,
- et de Cl_{objet} , la couleur propre de l'objet.



Justifications.

Preuve du lemme 1.

Démonstration. Notons \vec{p} un vecteur orthogonal à \vec{n} , appartenant au plan $(P, \vec{n}, \vec{\ell})$.



Dans la base (\vec{n}, \vec{p}) , écrivons :

$$\vec{\ell} = \alpha \vec{n} + \beta \vec{p} \quad (1)$$

Alors, par symétrie, on aura :

$$\vec{r} = \alpha \vec{n} - \beta \vec{p}.$$

Calcul de α . On effectue le produit scalaire des termes de l'égalité (1) avec \vec{n} :

$$\vec{\ell} \cdot \vec{n} = \alpha \vec{n} \cdot \vec{n} + \beta \vec{p} \cdot \vec{n},$$

donc

$$\vec{\ell} \cdot \vec{n} = \alpha \cdot 1 + \beta \cdot 0.$$

Ainsi $\alpha = \vec{\ell} \cdot \vec{n}$.

Calcul de \vec{r} . On repart de l'égalité (1) :

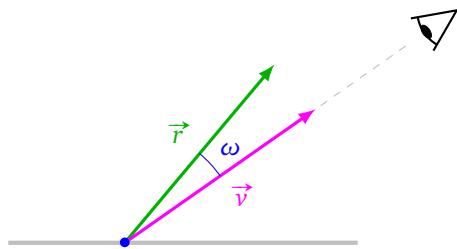
$$\beta \vec{p} = \vec{\ell} - \alpha \vec{n},$$

donc

$$\vec{r} = \alpha \vec{n} - \beta \vec{p} = \alpha \vec{n} - (\vec{\ell} - \alpha \vec{n}) = 2\alpha \vec{n} - \vec{\ell} = 2(\vec{\ell} \cdot \vec{n}) \vec{n} - \vec{\ell}.$$

□

Calcul de l'angle ω . Notons ω l'angle entre \vec{v} (dirigé vers l'observateur) et \vec{r} (axe principal de réflexion).



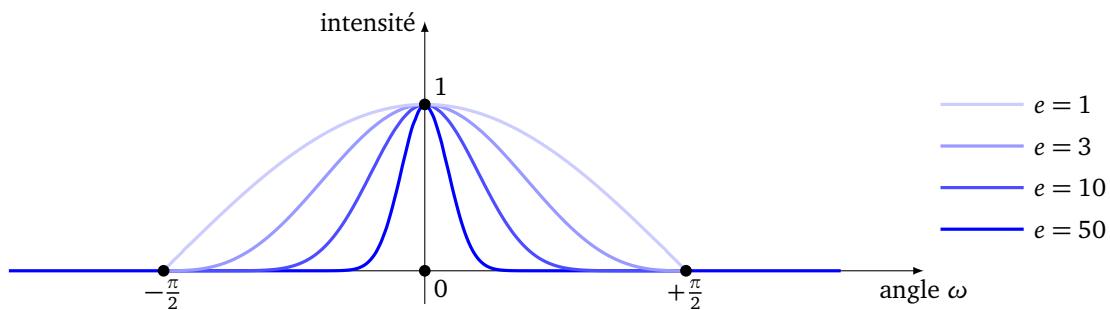
Alors :

$$\cos(\omega) = \vec{r} \cdot \vec{v}.$$

Intensité réfléchie/perçue. L'intensité renvoyée par réflexion dépend de l'angle ω , elle diminue lorsque l'observateur s'éloigne de l'axe principal de réflexion, selon la formule :

$$(\cos(\omega))^{e_{\text{spec}}}$$

Plus l'exposant est grand, plus l'intensité lumineuse se concentre autour de l'axe principal de réflexion.

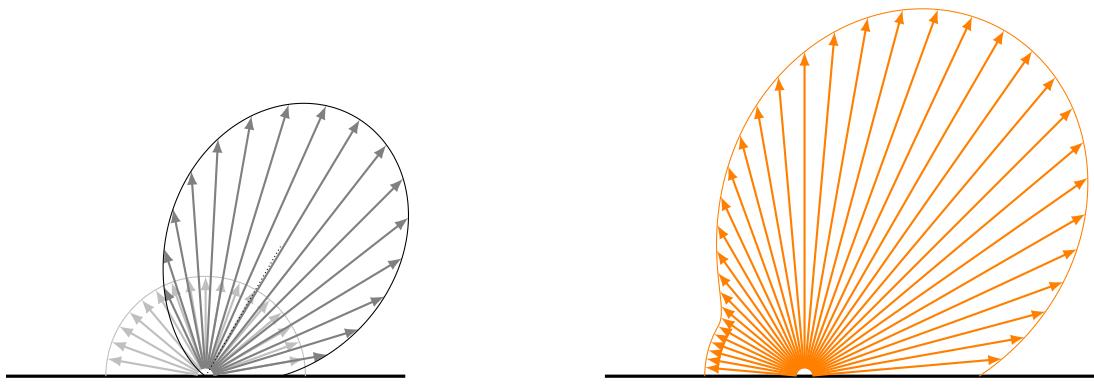


Validité des formules.

D'une part, on souhaite éclairer la face visible, donc il faut $\vec{\ell} \cdot \vec{n} \geq 0$. D'autre part, l'observateur doit être du « bon » côté de la réflexion, c'est-à-dire $\vec{r} \cdot \vec{v} \geq 0$. Dans les autres cas, la lumière atteint une face cachée ou bien n'est pas du tout renvoyée vers l'observateur.

Lumière diffuse/lumière spéculaire. Un même éclairage peut avoir une composante diffuse et une composante spéculaire, on distingue alors deux intensités $i_{\text{source,diff}}$ et $i_{\text{source,spec}}$.

Voici un exemple de combinaison de ces deux composantes : à gauche les luminosités diffuses et spéculaires séparées, à droite la lumière résultante.



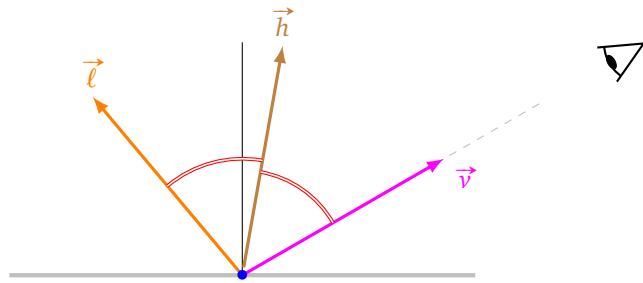
Variante Blinn–Phong.

Comme on l'a dit, la lumière spéculaire nécessite beaucoup de calculs car il faut à chaque fois calculer le vecteur \vec{v} qui dépend de C et de P , mais aussi le vecteur \vec{r} .

Il existe une variante pour la formule de la luminosité spéculaire, appelée **variante Blinn–Phong**. Certains trouvent le rendu un peu plus réaliste, et elle a l'avantage de demander moins de calculs dans certaines situations.

Définissons le **vecteur bissecteur** (*half-way vector*) :

$$\vec{h} = \frac{\vec{l} + \vec{v}}{\|\vec{l} + \vec{v}\|}.$$



La nouvelle formule est alors :

$$Cl'_{\text{spec}} = i_{\text{source}} (\vec{h} \cdot \vec{n})^{e'_{\text{spec}}} Cl_{\text{source}} \otimes_{cl} Cl_{\text{objet}}$$

valide lorsque $\vec{l} \cdot \vec{n} \geq 0$ et $\vec{h} \cdot \vec{n} \geq 0$.

Il faut ajuster l'exposant e'_{spec} pour retrouver des effets similaires à la formule classique.

Expliquons l'avantage de la formule de Blinn–Phong. Si on considère un éclairage directionnel, c'est-à-dire où \vec{l} est un vecteur constant et un observateur éloigné, c'est-à-dire \vec{v} est aussi un vecteur constant, alors le vecteur bissecteur \vec{h} est un vecteur constant (indépendant de l'objet) et n'a besoin d'être calculé qu'une seule fois, pour calculer le produit scalaire $\vec{h} \cdot \vec{n}$. Par contre dans la formule classique, le calcul de $\vec{r} \cdot \vec{n}$ nécessite au préalable le calcul de \vec{r} , qui varie pour chaque surface élémentaire (via \vec{n}).

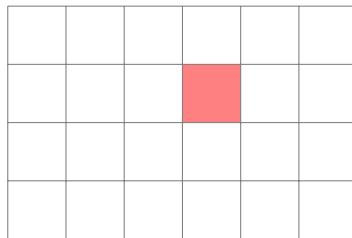
Pixels

Comment tracer, pixel par pixel, les figures géométriques de base ?

1. Pixels

1.1. L'unité d'affichage

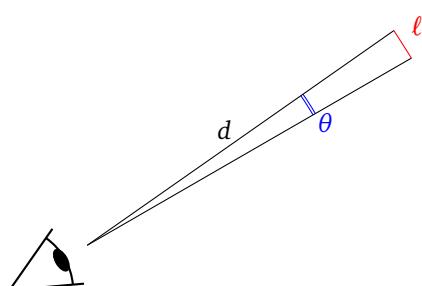
Le **pixel** est l'unité élémentaire d'affichage graphique numérique. Son abréviation est « px ». Le plus souvent le pixel est assimilé à un carré (ce sera le cas dans ce cours), mais peut aussi parfois être un rectangle.



Un pixel peut être noir ou blanc, mais aussi en couleur (le plus souvent obtenue par l'addition des couleurs de trois sous-pixels : un rouge, un vert, un bleu). La taille d'un pixel est généralement inférieure à 1 mm, elle dépend du type d'écran. Plus la taille d'un pixel est petite, plus la résolution est bonne. Une autre unité de mesure de la résolution est le **nombre de pixels par pouce**, abrégé par « ppp » (ou *ppi* pour *pixels per inch*). Comme son nom l'indique, il s'agit de compter combien de pixels on peut juxtaposer sur une longueur d'un pouce (1 pouce = 25,4 mm). L'équivalent dans le monde de l'impression est le *dpi* (*dots per inch*). Voici quelques exemples approximatifs :

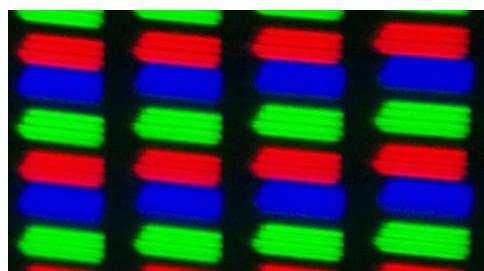
Type d'écran	Taille du pixel (mm)	Résolution (ppp)
Télévision	0.5	50
Ordinateur portable	0.17	150
Téléphone	0.05	500

L'œil humain a une acuité d'environ 1 minute d'arc (c'est-à-dire $1/60$ de degré, soit $\theta = \frac{1}{60} \frac{2\pi}{360} \simeq 0.00029$ radian). Ainsi à une distance d , l'œil distingue des pixels d'une taille $\ell \simeq d\theta$ (avec ℓ et d en mètre et θ en radian).



Ainsi en regardant un écran à 20 cm de distance, augmenter la résolution au-delà de 400 ppp ne permet de ressentir une amélioration. Il faut alors mieux concentrer ses efforts sur la qualité de l'image (vitesse de rafraîchissement, couleurs, rayonnements...) et sur l'énergie (consommation, encre électronique...).

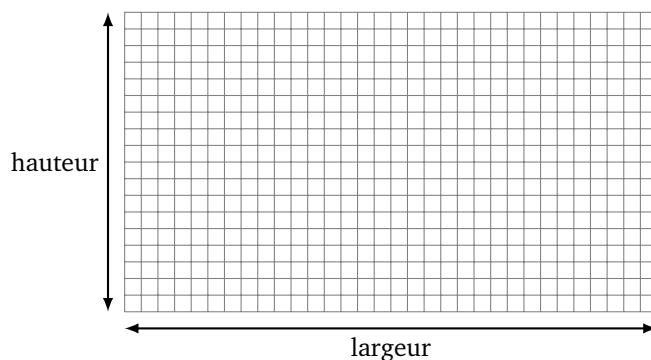
Un pixel n'est pas vraiment un petit carré ! Dans ce cours nous assimilerons un pixel à un (petit) carré. Cependant la réalité est plus compliquée. Un écran est composé d'une grille de sources lumineuses, appelées des *sous-pixels*. Chaque source illumine une petite zone qui n'est pas nécessairement un carré (cela peut être un disque ou un rectangle). Chaque source est souvent d'une seule couleur parmi rouge/vert/bleu, mais son intensité peut varier. Il n'y a donc physiquement aucun carré dans ce processus ! Sur la photo ci-dessous on pourrait regrouper trois sous-pixels superposés et appeler le petit carré correspondant un pixel, en lui attribuant la couleur résultant des trois sous-pixels. Il est plus juste de dire que les pixels sont un échantillonnage (relevé en des points d'une grille) du signal lumineux produit par l'écran.



Les sous-pixels d'un écran de téléphone portable (photo *Wikimedia*).

1.2. Écran

La **taille** d'un écran est le couple (largeur, hauteur) exprimé en pixels, habituellement notée sous la forme largeur \times hauteur. Le résultat du produit correspond au nombre total de pixels de l'écran.



Le **rapport d'image** est le quotient de la largeur par la hauteur :

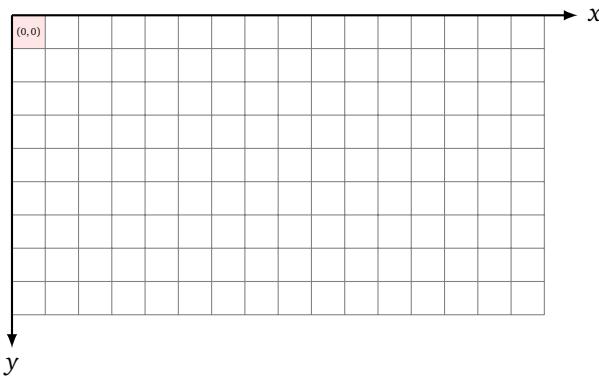
$$\text{rapport d'image} = \frac{\text{largeur}}{\text{hauteur}}.$$

Par exemple, pour un écran de taille 1024×768 , cela signifie que chaque ligne contient 1 024 pixels et que chaque colonne contient 728 pixels. Le rapport d'image est

$$r = \frac{1024}{768} = \frac{4}{3} \simeq 1.33.$$

Nom du format	largeur	hauteur	rapport (fraction)	ratio (approché)
VGA	640	480	4/3	1.33
HD720	1280	720	16/9	1.77
HD1080	1920	1080	16/9	1.77
WUXGA	1920	1200	16/10	1.60
4K	3840	2160	16/9	1.77
8K	7680	4320	16/9	1.77
Image format Cinémascope	1024	430		2.38

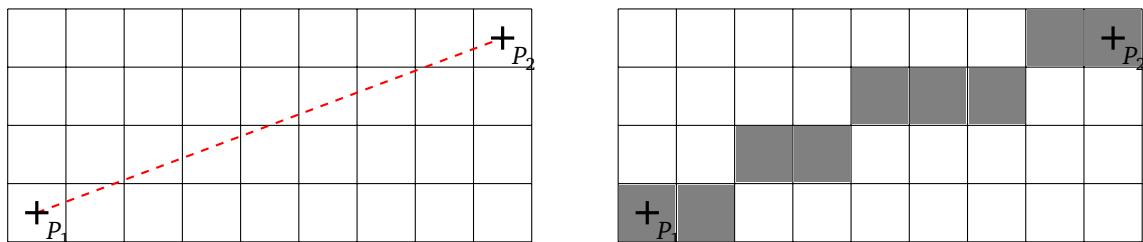
Le repère des pixels sur un écran a son axe des « y » dirigé vers le bas, contrairement à l'usage en mathématiques (que l'on suivra dans cet ouvrage). Ainsi le pixel en haut à gauche a pour coordonnées $(0, 0)$.



2. Tracé de droites et de cercles

2.1. Tracé élémentaire d'un segment

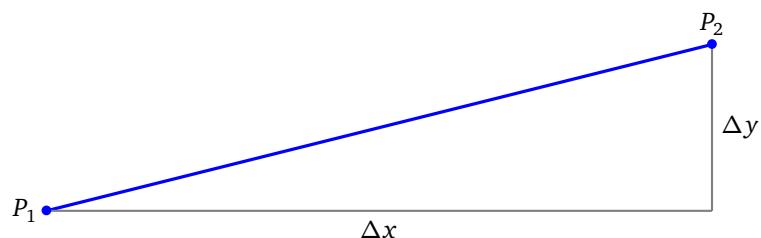
Le but est de tracer un segment composé de pixels entre deux points $P_1(x_1, y_1)$ et $P_2(x_2, y_2)$. Nous supposons que ces points ont des coordonnées entières ($x_1, y_1, x_2, y_2 \in \mathbb{Z}$) et qu'un pixel est un carré de côté 1 centré en un couple d'entier (i, j) .



L'idée naturelle est de calculer une équation de la droite (P_1P_2). Une équation réduite est $y = \alpha x + \beta$ où α est la pente et β l'ordonnée à l'origine. On va préférer écrire l'équation sous la forme :

$$y = \alpha(x - x_1) + y_1$$

où (x_1, y_1) est un point de la droite et la pente se calcule par $\alpha = \frac{\Delta y}{\Delta x} = \frac{y_2 - y_1}{x_2 - x_1}$.



Hypothèse. On suppose $x_1 < x_2$, $y_1 < y_2$ et $\alpha = \frac{\Delta y}{\Delta x} \leq 1$.

Autrement dit, on trace un segment ayant une pente comprise entre 0 et 1, c'est-à-dire l'angle formé entre une horizontale et le segment est compris entre 0° et 45° . Les autres situations s'obtiennent par symétrie et ne seront pas détaillées ici.

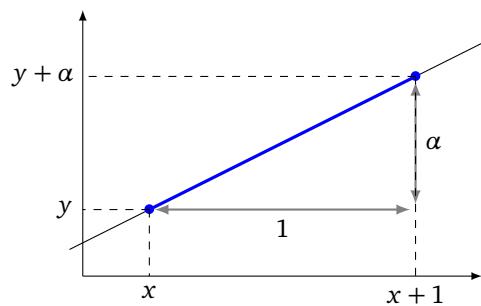
Les pixels à colorier ont pour abscisses successives $i = x_1, x_1 + 1, \dots, x_2$. L'ordonnée réelle correspondant à l'abscisse i est $y = \alpha(i - x_1) + y_1$ mais nous voulons une coordonnée entière qui s'obtient par un arrondi :

$$j = \text{arrondi}(\alpha(i - x_1) + y_1)$$

Remarque : si $\lfloor x \rfloor$ désigne la partie entière d'un réel x (obtenue par une commande `floor(x)` en Python par exemple) alors :

$$\text{arrondi}(x) = \lfloor x + \frac{1}{2} \rfloor$$

Pour minimiser les opérations on remarque que lorsque l'on passe de l'abscisse x à l'abscisse $x + 1$ alors l'ordonnée passe de y à $y + \alpha$.



Algorithme (Algorithme du tracé élémentaire).

Entrée : des entiers x_1, y_1, x_2, y_2 vérifiant l'hypothèse.

Sortie : le tracé des pixels reliant (x_1, y_1) à (x_2, y_2) .

- Calculer $\alpha = \frac{y_2 - y_1}{x_2 - x_1}$.
- Poser $i = x_1$.
- Poser $y = y_1$.
- Tant que $i \leq x_2$:
 - $j = \text{arrondi}(y)$
 - afficher le pixel (i, j)
 - $i = i + 1$
 - $y = y + \alpha$

Exemple.

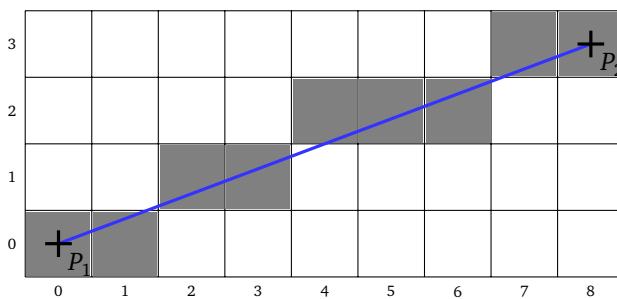
On souhaite tracer le segment reliant $P_1(0, 0)$ et $P_2(8, 3)$. On calcule :

$$\alpha = \frac{\Delta y}{\Delta x} = \frac{y_2 - y_1}{x_2 - x_1} = \frac{3}{8} = 0.375.$$

Voici les valeurs lorsqu'on applique l'algorithme :

i	y	j
0	0	0
1	0.375	0
2	0.75	1
3	1.125	1
4	1.5	2
5	1.875	2
6	2.25	2
7	2.625	3
8	3.0	3

Ce qui donne :

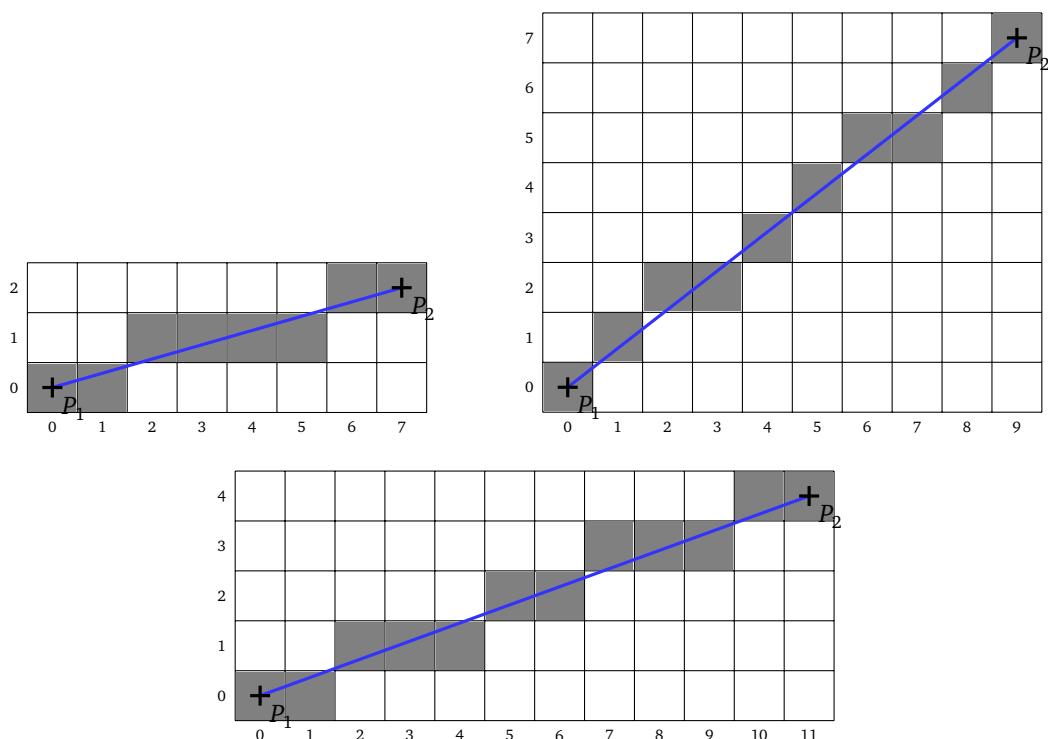


Noter que les calculs se font avec des nombres réels et sont donc assez lents.

2.2. Algorithme de Bresenham

Objectif : trouver un algorithme plus rapide que l'algorithme élémentaire en ne faisant que des calculs sur des entiers. Le résultat est équivalent à l'algorithme précédent mais beaucoup plus rapide.

Voici quelques tracés.



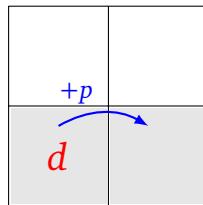
Algorithme (Algorithme de Bresenham).

Entrée : des entiers x_1, y_1, x_2, y_2 vérifiant l'hypothèse.

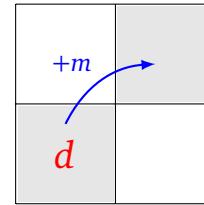
Sortie : le tracé des pixels reliant (x_1, y_1) à (x_2, y_2) .

- Calculer $p = 2\Delta y = 2(y_2 - y_1)$.
- Calculer $m = 2\Delta y - 2\Delta x = 2(y_2 - y_1) - 2(x_2 - x_1)$.
- Poser $d = 2\Delta y - \Delta x = 2(y_2 - y_1) - (x_2 - x_1)$.
- Poser $i = x_1$ et $j = y_1$.
- Tant que $i \leq x_2$:
 - afficher le pixel (i, j)
 - si $d < 0$:
 - $d = d + p$
 - sinon :
 - $j = j + 1$
 - $d = d + m$
 - $i = i + 1$

Noter que p, m sont des constantes entières avec $p > 0$ et $m < 0$; d est le « défaut », c'est une variable qui détermine s'il faut prendre le pixel juste à côté ou bien s'il faut monter d'un cran. Le défaut est alors mis à jour en lui ajoutant selon le cas une valeur fixe $p > 0$ ou $m < 0$ (si $d < 0$, le nouveau défaut $d + p$ a augmenté; si $d \geq 0$, le nouveau défaut $d + m$ a diminué car $m < 0$).



Cas $d < 0$.



Cas $d \geq 0$.

Noter qu'en plus les opérations sur les entiers sont élémentaires : ce sont des additions et un test de positivité.

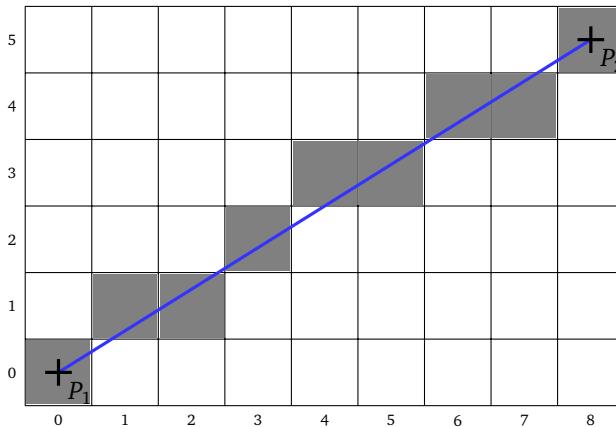
Exemple.

On souhaite tracer le segment reliant $P_1(0, 0)$ et $P_2(8, 5)$. On calcule :

$$p = 10 \quad m = -6 \quad d_0 = 2$$

Voici les valeurs lorsqu'on applique l'algorithme :

i	j	d	monter ?	nouvelle valeur de d
0	0	2	oui	-4
1	1	-4	non	6
2	1	6	oui	0
3	2	0	oui	-6
4	3	-6	non	4
5	3	4	oui	-2
6	4	-2	non	8
7	4	8	oui	2
8	5			



Preuve. Justifions que l'algorithme est correct.

- **Équation de la droite à coefficients entiers.** Nous avons vu qu'une équation de la droite (P_1P_2) est $y = \frac{\Delta y}{\Delta x}(x - x_1) + y_1$ avec $\Delta x = x_2 - x_1$ et $\Delta y = y_2 - y_1$. En multipliant cette équation par l'entier Δx on obtient l'équation :

$$(\Delta y)x - (\Delta x)y - (\Delta y)x_1 + (\Delta x)y_1 = 0$$

c'est-à-dire une équation $ax + by + c = 0$ avec a, b, c qui sont tous les trois des entiers.

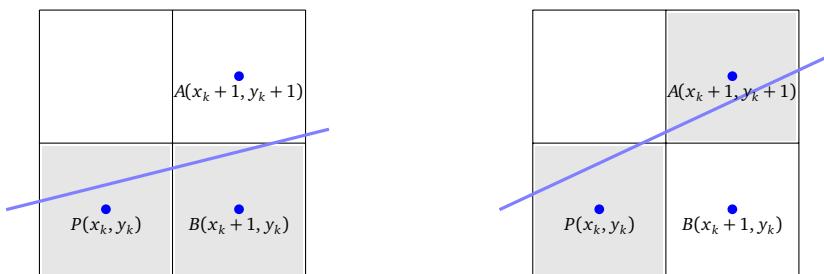
Ainsi on définit la fonction :

$$E(x, y) = ax + by + c$$

où $a = \Delta y$, $b = -\Delta x$, $c = -(\Delta y)x_1 + (\Delta x)y_1$. Cette fonction détermine l'écart du point (x, y) par rapport à la droite (P_1P_2) :

- si $E(x, y) = 0$, le point (x, y) appartient à la droite,
- si $E(x, y) < 0$, le point (x, y) est situé au-dessus de la droite,
- si $E(x, y) > 0$, le point (x, y) est situé en-dessous de la droite.

- **Monter ou pas?** Centrons en $P(x_k, y_k)$ un pixel déjà correctement colorié. Quels choix avons-nous pour colorier le pixel suivant ? Dans tous les cas le pixel suivant sera sur la colonne d'abscisse $x_k + 1$. Mais son ordonnée sera soit $y_k + 1$ (point A), soit y_k (point B).



Cas $D_k < 0$.

Cas $D_k \geq 0$.

Nous allons utiliser notre fonction d'écart E . Noter que l'on a toujours $E(A) \leq 0$ et $E(B) > 0$. On choisit entre A et B celui qui est le plus proche de la droite réelle, c'est-à-dire celui qui a le plus petit écart $|E(A)|$ ou $|E(B)|$ (en valeur absolue). Pour savoir lequel choisir on définit le *défaut* par

$$D(P) = E(A) + E(B).$$

On retient :

Si $D(P) \leq 0$ alors on choisit B, si $D(P) > 0$ alors on choisit A.

En effet, si $D(P) \leq 0$ alors $E(A) + E(B) \leq 0$ donc $E(B) \leq -E(A)$ mais comme $E(A) \leq 0$ et $E(B) > 0$ alors on a bien $|E(B)| \leq |E(A)|$.

- Calcul du défaut.** Calculons D_k le défaut au point $P(x_k, y_k)$:

$$\begin{aligned}
 D_k &= D(x_k, y_k) = E(A) + E(B) = E(x_k + 1, y_k + 1) + E(x_k + 1, y_k) \\
 &= a(x_k + 1) + b(y_k + 1) + c + a(x_k + 1) + b y_k + c \\
 &= 2ax_k + 2by_k + 2a + b + 2c
 \end{aligned}$$

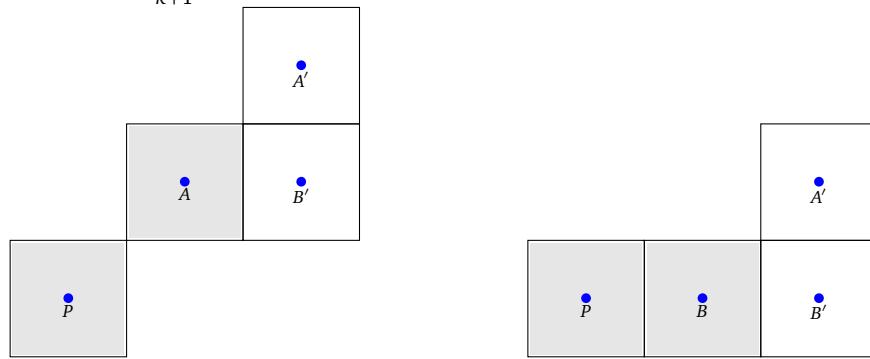
En particulier D_0 le défaut au point initial (x_1, y_1) est :

$$\begin{aligned}
 D_0 &= 2ax_1 + 2by_1 + 2a + b + 2c \\
 &= 2(ax_1 + by_1 + c) + 2a + b \quad (\star) \\
 &= 2a + b \\
 &= 2\Delta y - \Delta x
 \end{aligned}$$

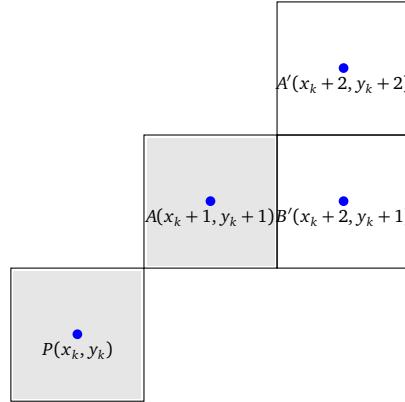
À la ligne (\star) on a utilisé le fait que le point initial (x_1, y_1) est exactement sur la droite, donc $ax_1 + by_1 + c = 0$.

- Formule de récurrence du défaut.**

Nous allons maintenant trouver la relation qui lie le défaut d'un pixel D_k au défaut du pixel suivant D_{k+1} . On part toujours du point $P(x_k, y_k)$, on a déjà calculé D_k . Le calcul de D_{k+1} dépend du choix A ou B pour le pixel d'abscisse x_{k+1} .



Cas du choix A (cas $D_k \geq 0$). Le pixel suivant a pour abscisse $x_k + 2$ et pour ordonnée soit $y_k + 2$ (A'), soit $y_k + 1$ (B').



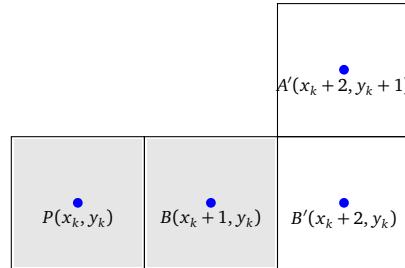
Calculons alors le défaut :

$$\begin{aligned}
 D_{k+1} &= E(A') + E(B') = E(x_k + 2, y_k + 2) + E(x_k + 2, y_k + 1) \\
 &= a(x_k + 2) + b(y_k + 2) + c + a(x_k + 2) + b(y_k + 1) + c \\
 &= 2ax_k + 2by_k + 4a + 3b + 2c \\
 &= D_k + 2a + 2b
 \end{aligned}$$

Ainsi en posant $m = 2a + 2b = 2\Delta y - 2\Delta x$, on obtient la formule de récurrence :

$$D_{k+1} = D_k + m.$$

Cas du choix B (cas $D_k < 0$). Le pixel suivant a pour abscisse $x_k + 2$ et pour ordonnée soit $y_k + 1$ (A'), soit y_k (B').



Cas B.

Calculons alors le défaut :

$$\begin{aligned} D_{k+1} &= E(A') + E(B') = E(x_k + 2, y_k + 1) + E(x_k + 2, y_k) \\ &= a(x_k + 2) + b(y_k + 1) + c + a(x_k + 2) + b y_k + c \\ &= 2ax_k + 2by_k + 4a + b + 2c \\ &= D_k + 2a \end{aligned}$$

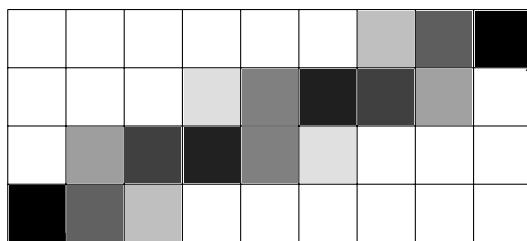
Ainsi en posant $p = 2a = 2\Delta y$, on obtient la formule de récurrence :

$$D_{k+1} = D_k + p.$$

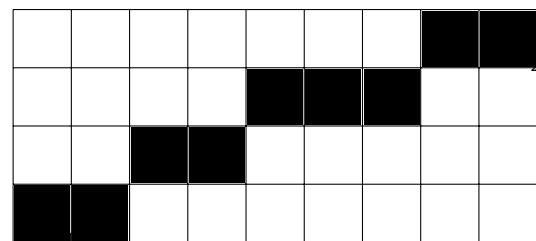
Conclusion : on sait calculer D_{k+1} qui permet de choisir entre A' et B' . Ainsi à partir de D_0 et par les formules de récurrence ci-dessus on calcule la suite des défauts D_k et on sait donc s'il faut monter le pixel d'un cran ou pas.

2.3. Anticrénelage (antialiasing)

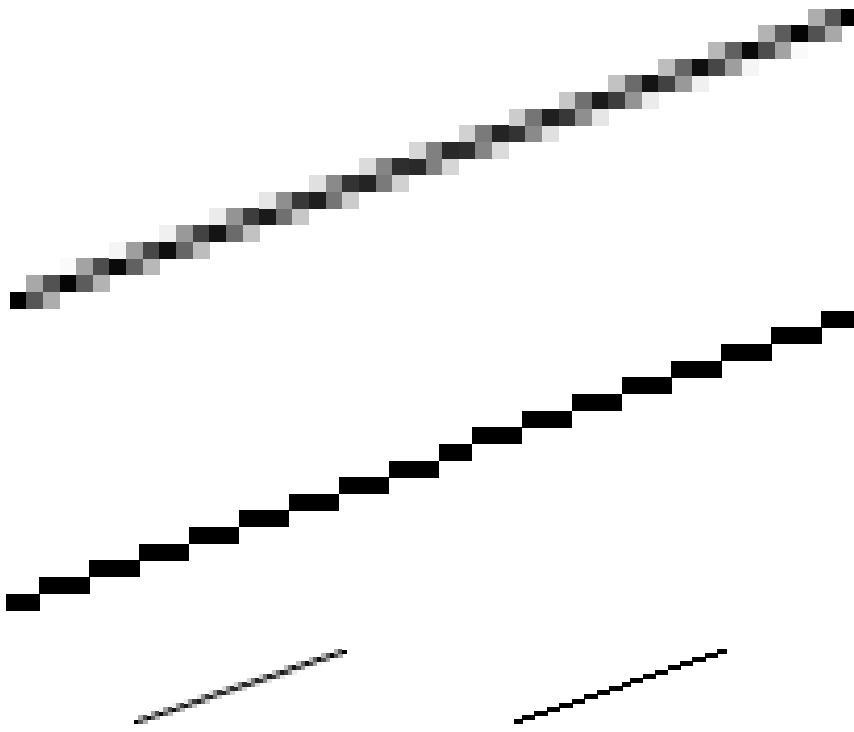
Le tracé d'un segment est nécessairement une approximation d'un tracé de droite, qui présente des défauts de visualisation appelée « crénélage ». On remédié à cette imperfection à l'aide d'un anticrénelage (*antialiasing*) qui colorie aussi le pixel laissé de côté (le pixel A ou B écarté dans les algorithmes précédents).



Avec anticrénelage



Sans anticrénelage



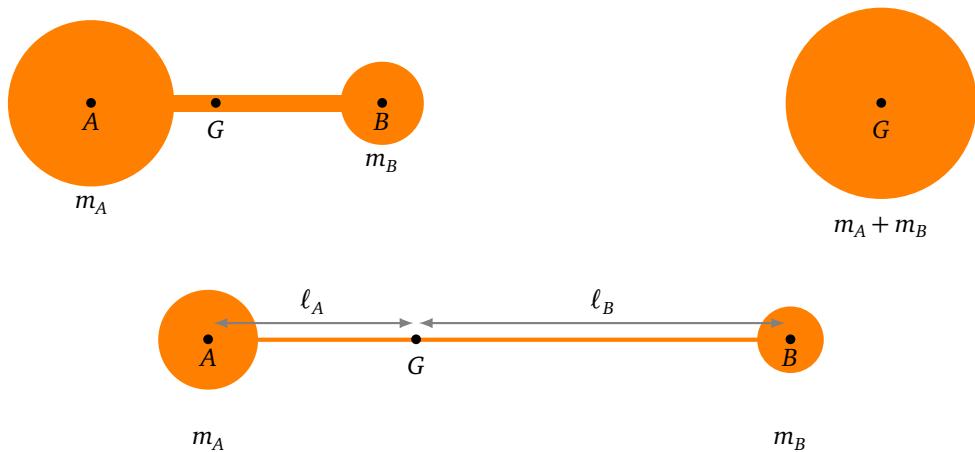
De près le rendu n'est pas impressionnant mais de loin l'amélioration est sensible.

Nous allons expliquer un algorithme simple pour dessiner un segment avec anticrénelage. Nous avons donc deux pixels l'un au-dessus de l'autre à colorier : le pixel A et le pixel B . Comment répartir 100% de couleur entre ces deux pixels ?

Commençons par expliquer une analogie issue de la physique. Considérons deux masses ponctuelles liées entre elles : m_A centrée en A et m_B centrée en B . Lorsque l'on regarde ce système d'assez loin on peut le considérer comme un système formé d'une seule masse ponctuelle : la masse totale est $m = m_A + m_B$ et est centrée au centre de gravité G . Ce centre de gravité G est défini par la relation :

$$\ell_A m_A = \ell_B m_B$$

où $\ell_A = AG$ et $\ell_B = BG$.



Reprendons cette idée : nous répartissons une masse totale $m = 1$ (100% de la couleur) centrée exactement sur la droite réelle en G entre deux points A et B .

Notons (x, y) le point G de la droite réelle. Soit alors :

$$j = \lfloor y \rfloor \quad \text{et} \quad \ell = y - \lfloor y \rfloor$$

respectivement la partie entière et la partie fractionnaire de y . La distance BG est ℓ et la distance AG est $1 - \ell$ (car $AB = 1$). Ainsi pour que les couleurs satisfassent la formule du centre de gravité il faut choisir $m_A = \ell$ et $m_B = 1 - \ell$ comme intensité de couleur respectivement pour A et B car on a alors bien $(1 - \ell)\ell = \ell(1 - \ell)$. On adapte l'algorithme du tracé élémentaire pour y ajouter l'anticrénelage.

Algorithme (Algorithme d'anticrénelage).

Entrée : des entiers x_1, y_1, x_2, y_2 vérifiant l'hypothèse.

Sortie : le tracé des pixels reliant (x_1, y_1) à (x_2, y_2) .

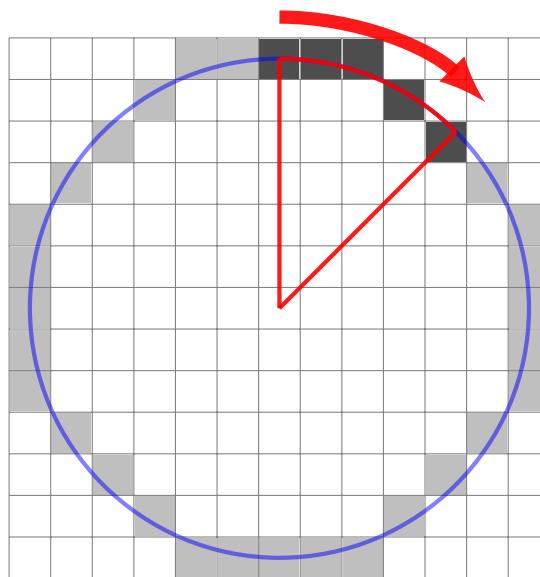
- Calculer $\alpha = \frac{y_2 - y_1}{x_2 - x_1}$.
- Poser $i = x_1$.
- Poser $y = y_1$.
- Tant que $i \leq x_2$:
 - $j = \lfloor y \rfloor$
 - $\ell = y - \lfloor y \rfloor$
 - afficher le pixel (i, j) avec l'intensité de couleur $1 - \ell$
 - afficher le pixel $(i, j + 1)$ avec l'intensité de couleur ℓ
 - $i = i + 1$
 - $y = y + \alpha$

2.4. Tracé d'un cercle

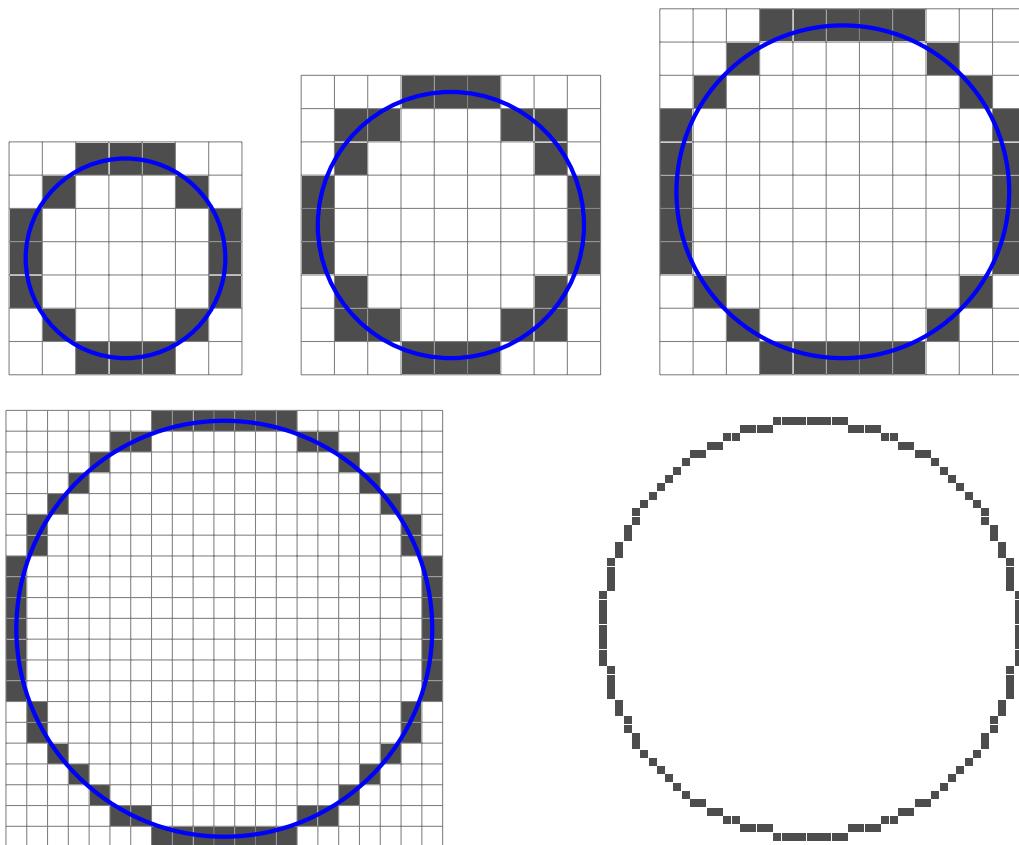
On souhaite tracer un cercle de rayon r centré en (x_0, y_0) . Ce cercle a pour équation :

$$(x - x_0)^2 + (y - y_0)^2 = r^2.$$

Pour cela on va se contenter de tracer l'arc de cercle compris entre 90° et 45° . Le reste s'obtiendra par symétrie.



Voici quelques exemples.



Voici l'algorithme du tracé de l'arc de cercle de type Bresenham car il n'utilise que des entiers. Le point de départ est le point $(0, r)$ (point le plus haut du cercle).

Algorithme (Algorithme de tracé de cercle).

Entrée : des entiers x_0, y_0 pour le centre et un entier r pour le rayon.

Sortie : le tracé d'un arc du cercle centré en (x_0, y_0) de rayon r .

- Poser $i = 0$.
- Poser $j = r$.
- Poser $d = 3 - 2r$.
- Tant que $i \leq j$:
 - afficher le pixel $(x_0 + i, y_0 + j)$
 - si $d < 0$:
 - $d = d + 4i + 6$
 - sinon :
 - $d = d + 4i - 4j + 10$
 - $j = j - 1$
 - $i = i + 1$

Sur la portion tracée (de 90° à 45°) il y a exactement un pixel pour chaque i . Pour obtenir le cercle complet, en plus du pixel en $(x_0 + i, y_0 + j)$, il faut afficher par symétrie les pixels en $(x_0 + j, y_0 + i)$, $(x_0 - i, y_0 + j)$, $(x_0 - j, y_0 + i)$, $(x_0 + i, y_0 - j)$, $(x_0 + j, y_0 - i)$, $(x_0 - i, y_0 - j)$, $(x_0 - j, y_0 - i)$.

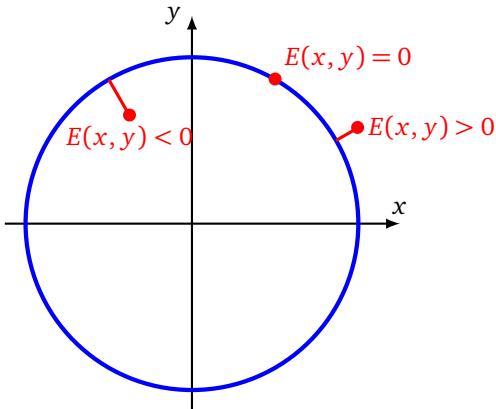
Preuve. La preuve est similaire à celle de l'algorithme du tracé d'un segment de Bresenham.

- **Équation du cercle.**

On suppose que le cercle est centré à l'origine : $(x_0, y_0) = (0, 0)$. Un cercle quelconque s'obtiendrait par translation. Soit la fonction d'écart :

$$E(x, y) = x^2 + y^2 - r^2$$

Si $E(x, y) = 0$ le point (x, y) est sur le cercle, si $E(x, y) > 0$ le point est à l'extérieur, si $E(x, y) < 0$ le point est à l'intérieur.



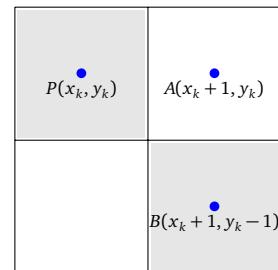
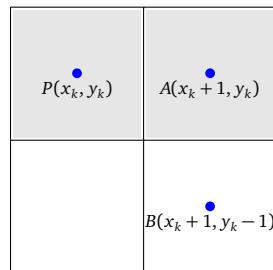
- **Descendre ou pas ?**

Pour un pixel centré en P déjà tracé, nous devons choisir pour le pixel suivant entre A et B . On choisit le point le plus proche du cercle, c'est-à-dire le minimum entre $|E(A)|$ ou $|E(B)|$. On définit le *défaut* par

$$D(P) = E(A) + E(B)$$

et on a :

Si $D(P) < 0$ alors on choisit A , si $D(P) \geq 0$ alors on choisit B .



Cas $D_k < 0$.

Cas $D_k \geq 0$.

- **Calcul du défaut.** Calculons D_k le défaut au point $P(x_k, y_k)$:

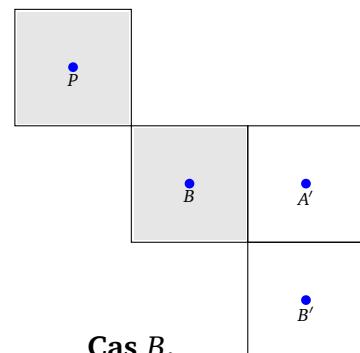
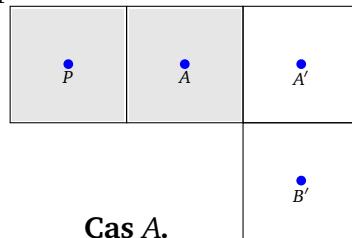
$$\begin{aligned} D_k &= D(x_k, y_k) = E(A) + E(B) = E(x_k + 1, y_k) + E(x_k + 1, y_k - 1) \\ &= (x_k + 1)^2 + y_k^2 - r^2 + (x_k + 1)^2 + (y_k - 1)^2 - r^2 \\ &= 2x_k^2 + 2y_k^2 + 4x_k - 2y_k + 3 - 2r^2 \end{aligned}$$

En particulier D_0 le défaut au point initial $(0, r)$ vaut :

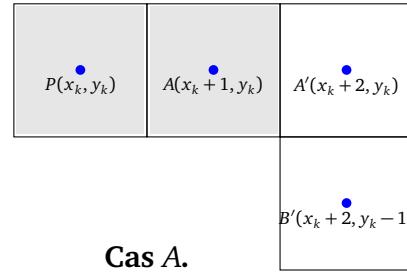
$$D_0 = 3 - 2r.$$

- **Formule de récurrence du défaut.**

On part du point $P(x_k, y_k)$, connaissant D_k on calcule D_{k+1} en fonction du choix A ou B pour le pixel d'abscisse x_{k+1} .



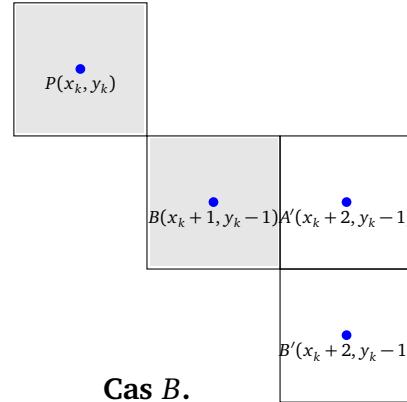
Cas du choix A (cas $D_k < 0$). Le pixel suivant a pour abscisse $x_k + 2$ et pour ordonnée soit y_k (A'), soit $y_k - 1$ (B').



Le défaut se calcule ainsi :

$$\begin{aligned} D_{k+1} &= E(A') + E(B') = E(x_k + 2, y_k) + E(x_k + 2, y_k - 1) \\ &= (x_k + 2)^2 + y_k^2 - r^2 + (x_k + 2)^2 + (y_k - 1)^2 - r^2 \\ &= D_k + 4x_k + 6 \end{aligned}$$

Cas du choix B (cas $D_k \geq 0$). Le pixel suivant a pour abscisse $x_k + 2$ et pour ordonnée soit $y_k - 1$ (A'), soit $y_k - 2$ (B').



$$\begin{aligned} D_{k+1} &= E(A') + E(B') = E(x_k + 2, y_k - 1) + E(x_k + 2, y_k - 2) \\ &= (x_k + 2)^2 + (y_k - 1)^2 - r^2 + (x_k + 2)^2 + (y_k - 2)^2 - r^2 \\ &= D_k + 4x_k - 4y_k + 10 \end{aligned}$$

Conclusion : on sait calculer D_{k+1} qui permet de choisir entre A' et B' .

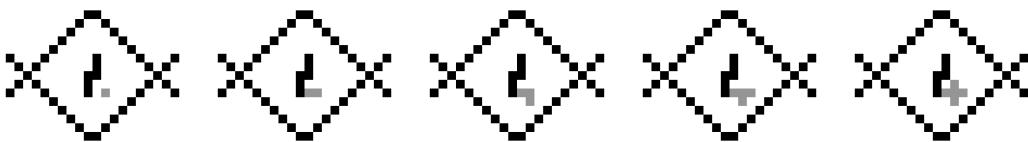
3. Colorier

3.1. Colorier par diffusion

Objectif. On considère une image formée par exemple de pixels noirs et blancs. On souhaite colorier en gris une zone de l'image. Pour cela, on part d'un pixel initial (la graine, *seed*) et il faut colorier en gris toute la zone de pixels blancs contenant la graine et bordée par des pixels noirs.



Principe. On part de la graine, qu'on colorie en gris. Pour chacun de ses quatre pixels voisins (gauche, bas, droite, haut), si c'est un pixel blanc, on applique le processus à partir de ce nouveau pixel.
Ci-dessous la graine et ses quatre voisins.



Mise en œuvre. Ce procédé peut être traduit en algorithmes par différentes méthodes : algorithme récursif, algorithme avec une pile ou algorithme avec une file. Dans tous les cas ce sont des algorithmes qui utilisent beaucoup de mémoire afin de stocker les pixels en attente d'être traités. On choisit un algorithme avec file (*queue ou fifo, first in, first out*) un peu moins gourmand en mémoire que les autres choix.

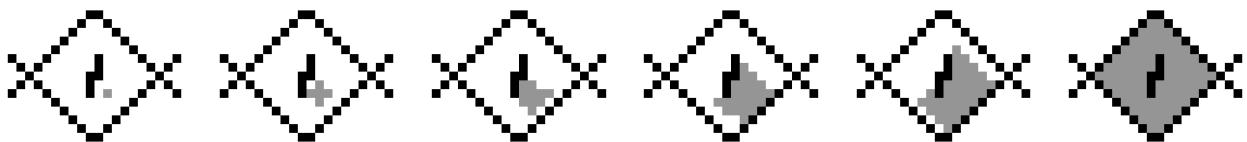
Algorithme (Algorithme de remplissage par diffusion (*flood fill*)).

Entrée : un tableau de pixels noirs ou blancs, un pixel initial (x_0, y_0) (la graine)

Sortie : un tableau de pixels noirs, blancs ou gris

- `file = [(x_0, y_0)]`
- Tant que la `file` n'est pas vide :
 - prendre (x, y) un élément de `file` (et le retirer de la file)
 - si (x, y) est un pixel blanc :
 - le colorier en gris,
 - ajouter à la `file` les quatre voisins $(x - 1, y)$, $(x, y + 1)$, $(x + 1, y)$, $(x, y - 1)$.

Quelques étapes du processus :



3.2. Colorier par balayage

Le coloriage par balayage est un peu plus performant au niveau de la mémoire car il colorie une portion de ligne avant d'ajouter les pixels au-dessus ou en-dessous dans la file d'attente.

Algorithme (Algorithme de balayage (*scan fill*)).

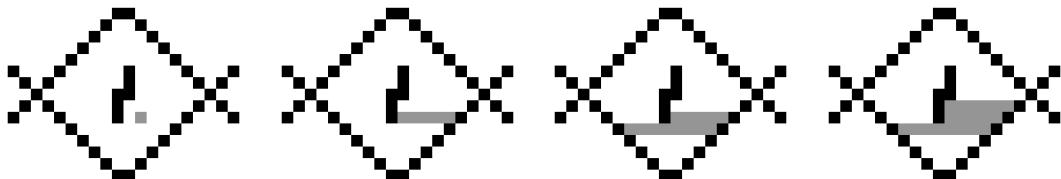
Entrée : un tableau de pixels noirs ou blancs, un pixel initial (x_0, y_0) (la graine)

Sortie : un tableau de pixels noirs, blancs ou gris

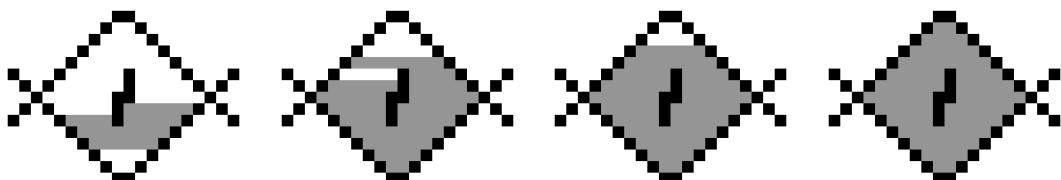
- `file = [(x_0, y_0)]`
- Tant que la `file` n'est pas vide :
 - prendre (x, y) un élément de `file` (et le retirer de la file)
 - poser $x_{\min} = x - 1$
 - tant que (x_{\min}, y) est un pixel blanc :
 - le colorier en gris,
 - faire $x_{\min} = x_{\min} - 1$.
 - poser $x_{\max} = x$
 - tant que (x_{\max}, y) est un pixel blanc :
 - le colorier en gris,
 - faire $x_{\max} = x_{\max} + 1$.
 - Pour chaque x' entre $x_{\min} + 1$ et $x_{\max} - 1$:

- si $(x', y + 1)$ est un pixel blanc, l'ajouter à la file,
- si $(x', y - 1)$ est un pixel blanc, l'ajouter à la file.

Ci-dessous la graine et les premières lignes.



Quelques étapes plus avancées :



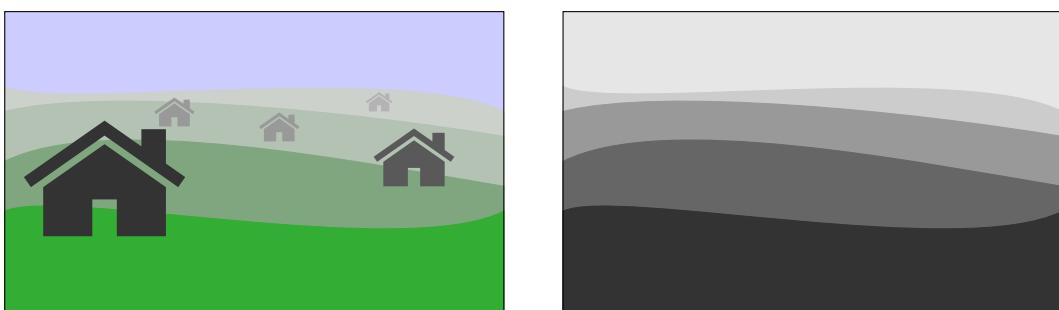
4. Animation

Nous expliquons quelques principes liés à l'animation des images. Les premiers dessins animés étaient basés sur des *calques* : une image fixe pour le calque de l'arrière-plan, un calque pour le premier plan avec par exemple un personnage à redessiner de nombreuses fois afin de simuler le mouvement et éventuellement des calques intermédiaires. On retrouvait aussi cette technique au cinéma où certains arrières-plans étaient peints.

4.1. Perspective atmosphérique

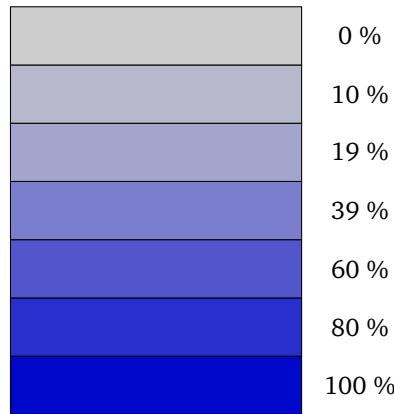
Voyons comment la technique des calques permet de simuler un paysage.

La **perspective atmosphérique** est une façon de représenter l'éloignement des différents plans ou calques par des teintes plus claires, des contrastes moins élevés et des couleurs plus grises (la saturation diminue). Cela reflète des phénomènes physiques réels, certaines particules de l'atmosphère (molécules d'air, gouttelettes d'eau, fumées...) diffractent la lumière (particulièrement les longueurs d'ondes plus courtes comme le bleu).



Il existe un modèle de couleurs HSB (*Hue/Saturation/Value* : Teinte/Saturation/Luminosité) pour lequel il suffit donc de changer le paramètre de saturation.

Ci-dessous les couleurs obtenues pour (h, s, b) avec $h = 0.66$, $b = 0.8$ et la saturation s variant de 0 à 1.



4.2. La 2.5D

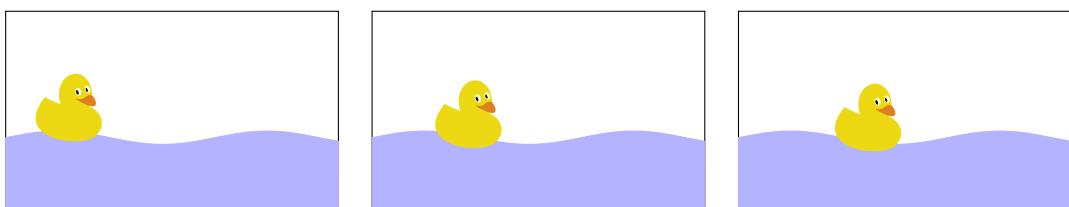
Comme son nom le suggère, la 2.5D est un ensemble des techniques d'animation qui permettent de simuler une scène 3D. C'est le cas de beaucoup de jeux vidéos de stratégie où le point de vue du joueur est au-dessus du plateau de jeu. Les objets représentés en perspective (par exemple ci-dessous les immeubles) sont des petites images 2D pré-dessinées. C'est une technique efficace mais qui nécessite de fixer ou de limiter fortement le choix de la vue du joueur.



SimCity 2000 (images : Wikipedia)

4.3. Principe de l'animation 2D

Vous le savez, on simule le mouvement à l'écran par un succession rapide d'images. Ainsi les pixels ne se déplacent pas, mais seule leur couleur change. Le cerveau humain, entraîné par les objets réels en mouvement, reconstitue un déplacement à partir d'images fixes.



Pour réaliser de petits jeux animés, les modules de *Pygame* de *Python* sont très sympas. On explique ci-dessous les principes de l'animation 2D en suivant la philosophie de *Pygame*.

Animation 1D.

Commençons par une animation très élémentaire en une seule dimension. Les objets en jeu sont :

- fond qui est l'image d'arrière-plan représentée ici par une liste de 0,

- un entier p qui représente la position d'un personnage, ce personnage est représenté par un 1,
- ecran qui est l'image finale affichée, et sera donc une juxtaposition de 0 et de 1.

Les étapes de l'animation sont les suivantes :

- *Initialisation*. On initialise ecran avec l'image d'arrière-plan fond :

00000000000

On définit une position de départ, par exemple $p = 3$.

- *Affichage*. Par $\text{ecran}[p] = 1$ on affiche le personnage :

0001000000

- *Déplacement*.

— *Effacer le personnage*. $\text{ecran}[p] = 0$

— *Changer la position*. $p = p + 1$

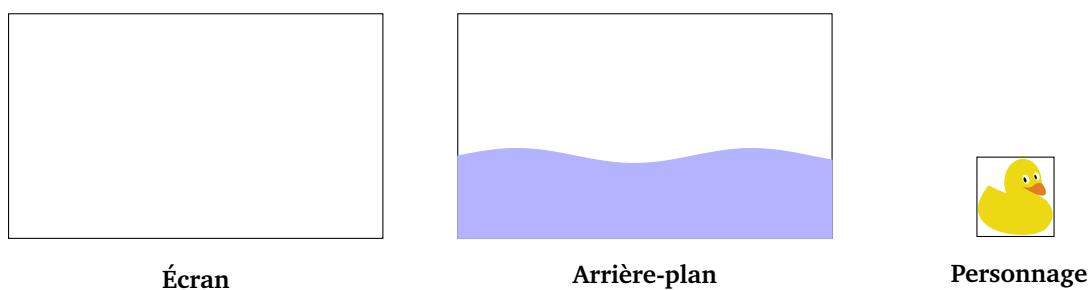
— *Réafficher le personnage*. $\text{ecran}[p] = 1$

En répétant le déplacement on obtient, successivement :

0001000000
0000100000
0000010000
0000001000
0000000100

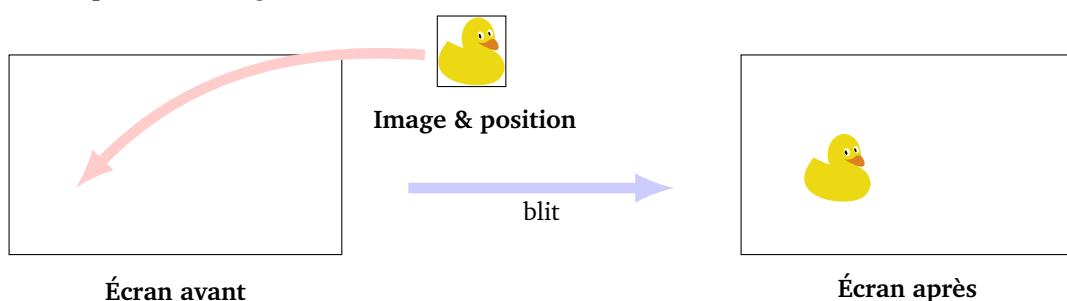
Animation 2D.

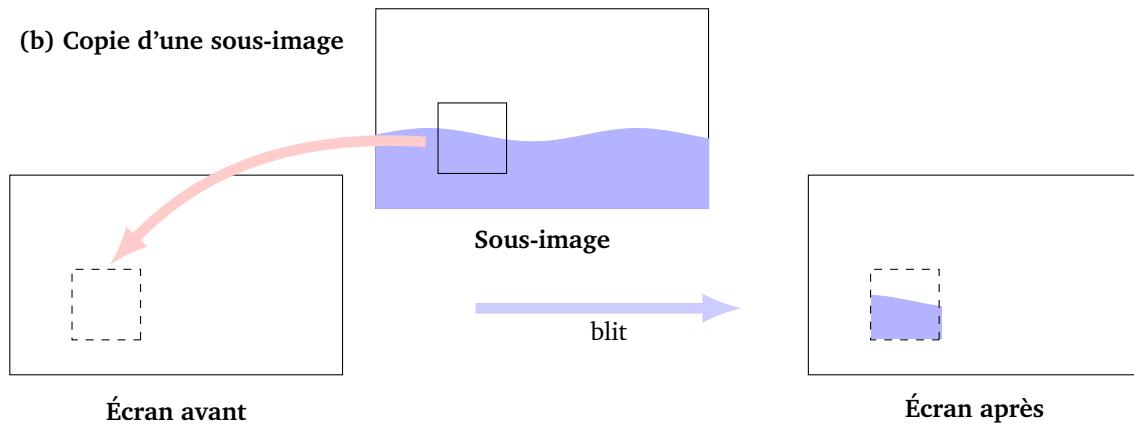
Le principe est similaire, il nous faut une image de fond, une image à afficher pour l'objet à déplacer. L'affichage à l'écran correspond à affecter des valeurs à la mémoire graphique (*screen buffer*) ; on représente ici l'écran comme une image.



L'opération fondamentale est la *copie rapide* ou *blit* (pour *bit-block transfer*) qui consiste à copier les données d'une image dans la mémoire graphique. Concrètement cela signifie copier une image et l'afficher à l'écran à une position donnée (figure (a)) ou bien copier seulement une partie d'une image (figure (b)).

(a) Copie d'une image



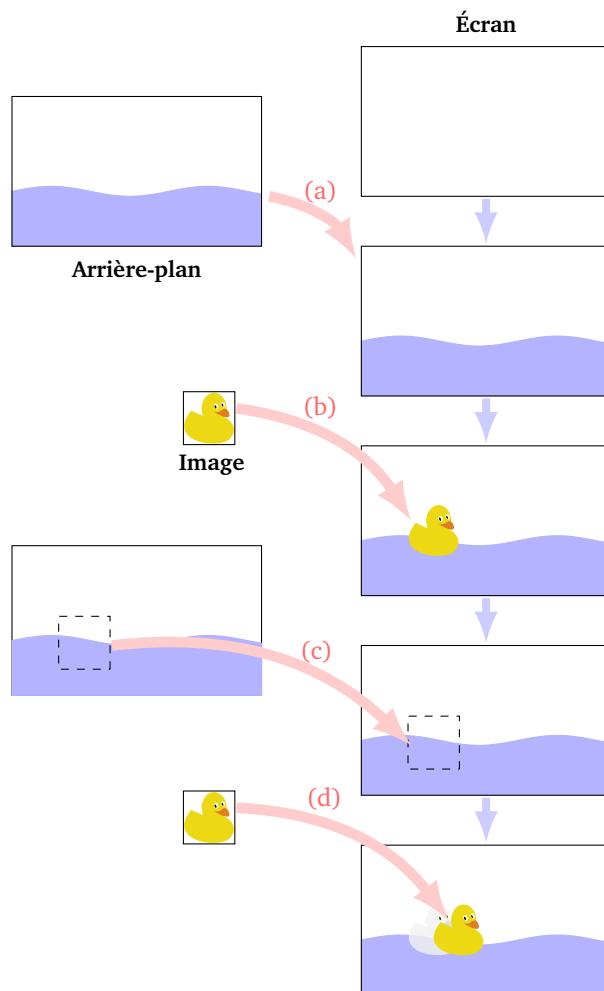


Le second cas de figure nous sert à effacer le personnage avant de le déplacer, plus exactement on copie à l'écran la portion de l'arrière-plan à la place qu'occupait le personnage.

Le processus de l'animation est alors le même que précédemment :

- (a) *Initialisation.* On initialise écran avec l'image d'arrière-plan fond :
 - (b) *Affichage.* On affiche le personnage à une position donnée.
 - *Déplacement.* (À répéter)
 - (c) *Effacement du personnage.* Pour cela on copie la portion de l'arrière-plan correspondant à l'emplacement qu'occupe le personnage.
 - (d) *Affichage du personnage à sa nouvelle position.*

La vitesse du mouvement est contrôlée par la longueur du déplacement et la durée de la pause entre deux répétitions.



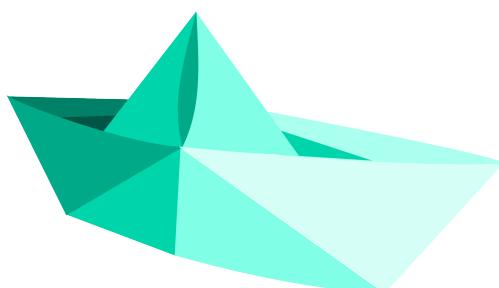
Texture

Les textures permettent de rendre les objets 3D beaucoup plus réalistes en simulant la couleur et la forme d'une matière. Il s'agit principalement de transformer un carré du plan en une surface de l'espace.

1. Motivation : texture en une dimension

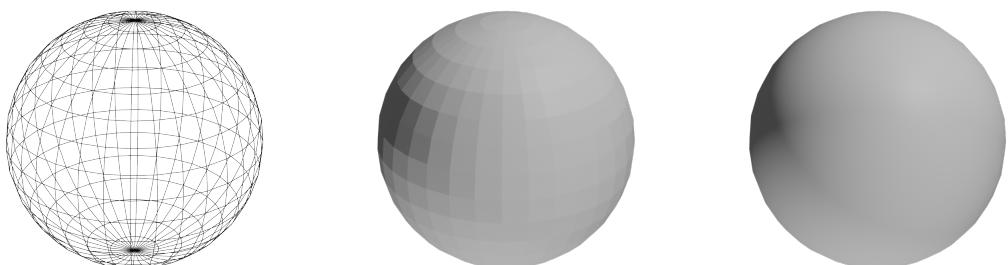
1.1. Motivation

L'objectif est de décorer un objet 3D déjà modélisé. Une analogie simple serait d'emballer cet objet dans un papier cadeau sans faire de plis. Ce n'est pas possible en général, c'est un théorème mathématique : par exemple on ne peut pas emballer parfaitement une sphère à l'aide d'une feuille de papier. Une variante de cet énoncé est qu'on ne peut pas représenter parfaitement le globe terrestre sur une carte ; il existe différents types de cartes du monde, mais elles déforment les longueurs, les aires ou bien les angles.



Cependant tout n'est pas perdu : à partir d'une feuille de papier, un tuto d'origami vous explique comment faire un petit bateau 3D. Si vous avez parfaitement anticipé les pliages et avez colorié votre feuille 2D intelligemment alors une fois formé votre bateau est déjà peint ! Heureusement pour nous pas besoin d'être ingénieur, nous allons considérer que notre objet 3D est déjà découpé en morceaux plat que l'on va décorer un par un.

Ci-dessous une sphère modélisée par un maillage (à gauche), avec un rendu plat (au centre), avec un rendu lisse (à droite).



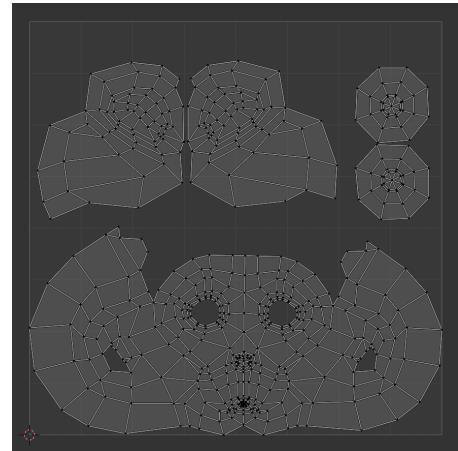
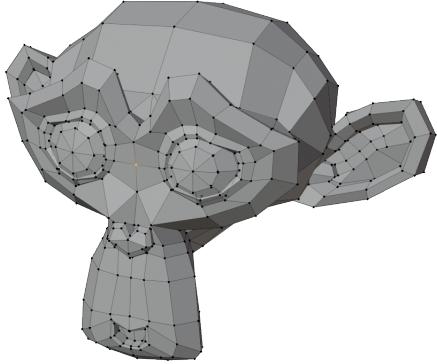
Voici trois images de texture :



Voici l'application de ces différentes textures :



Ci-dessous un objet 3D obtenu comme une union de faces (à gauche). Ce maillage est aplati (à droite) en ce qui s'appelle une *uv-map*, c'est sur ce patron 2D que la texture est dessinée avant d'être appliquée sur l'objet 3D.



Quel est l'intérêt d'une texture ? Tout d'abord cela permet une séparation conceptuelle et pratique entre deux tâches différentes : modéliser un objet d'une part et le décorer d'autre part. Le texturage lui-même peut se décomposer en plusieurs niveaux : le décor, mais aussi la matière, la granulosité... Ensuite il est plus facile pour un graphiste de créer une jolie texture en 2D que de décorer directement sur l'objet 3D. Enfin, une texture apporte généralement un gain de temps important pour les calculs. Est-il vraiment utile de modéliser chaque brique d'un mur ou chaque carré de pelouse surtout s'il s'agit d'un décor assez lointain ?

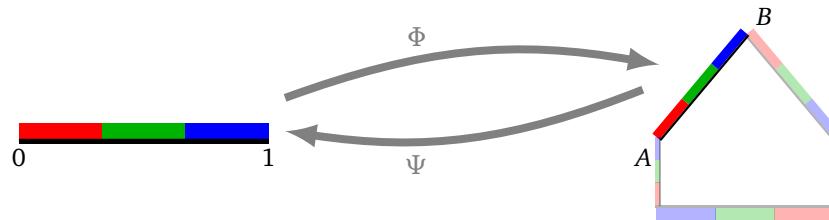
1.2. Texture (dimension 1)

L'idée pour appliquer une texture est simple : on part d'une image et on la colle sur un objet. Voyons ce que pourrait être une définition mathématique. Nous commençons à chaque fois les explications en une seule dimension. Même si une texture en dimension 1 a peu d'utilité, les concepts sont les mêmes que pour la dimension 2 mais les équations sont plus simples.

Une **texture** (en dimension 1) est une fonction :

$$\begin{aligned}\mathcal{C} : [0, 1] &\longrightarrow E \\ u &\longmapsto \mathcal{C}(u)\end{aligned}$$

Explications : on part de l'intervalle de référence $[0, 1]$, à chaque point de cet intervalle on associe une valeur appartenant à un ensemble E . Le cas le plus fréquent est d'associer une couleur $\mathcal{C}(u)$ à chaque point $u \in [0, 1]$, par exemple sous la forme d'une valeur $\mathcal{C}(u) = (r, g, b) \in [0, 1]^3$. Autrement dit, il s'agit de colorier chaque pixel de l'intervalle fixe $[0, 1]$.



Considérons maintenant un objet O et une de ses faces F . Si l'objet est dans le plan, une face F est un intervalle $[A, B]$ du plan. La **fonction de plaquage** est une application bijective Φ :

$$\begin{aligned}\Phi : [0, 1] &\longrightarrow F \\ u &\longmapsto \Phi(u)\end{aligned}$$

Nous notons $\Psi = \Phi^{-1}$ sa bijection réciproque, qui nous sera utile :

$$\begin{aligned}\Psi : F &\longrightarrow [0, 1] \\ P &\longmapsto \Psi(P)\end{aligned}$$

Appliquer une texture c'est l'action de $\mathcal{C} \circ \Psi$, c'est-à-dire associer à chaque P de la face F une couleur par la fonction :

$$\mathcal{C}(\Psi(P))$$

Autrement dit :

- pour déterminer la couleur de P , on calcule à quel paramètre $u \in [0, 1]$ il correspond : $u = \Psi(P)$. La couleur de P est celle de u : $\mathcal{C}(u)$. Cette façon de faire est le *texturage inverse* : on part du pixel de l'objet à colorier et on cherche sa couleur par l'application inverse Ψ .
- une autre façon de procéder est le *texturage direct* : pour chaque $u \in [0, 1]$ on calcule $P = \Phi(u)$ et on le colorie par la couleur $\mathcal{C}(u)$.

Quelques remarques. Le texturage ne se limite pas à associer une couleur, nous verrons des exemples d'autres situations un peu plus tard. On pourrait aussi considérer une face F qui ne soit pas un segment, un arc de cercle par exemple.

1.3. Texture (dimension 2)

Une **texture** (en dimension 2) est une fonction :

$$\begin{aligned}\mathcal{C} : [0, 1]^2 &\longrightarrow E \\ (u, v) &\longmapsto \mathcal{C}(u, v)\end{aligned}$$

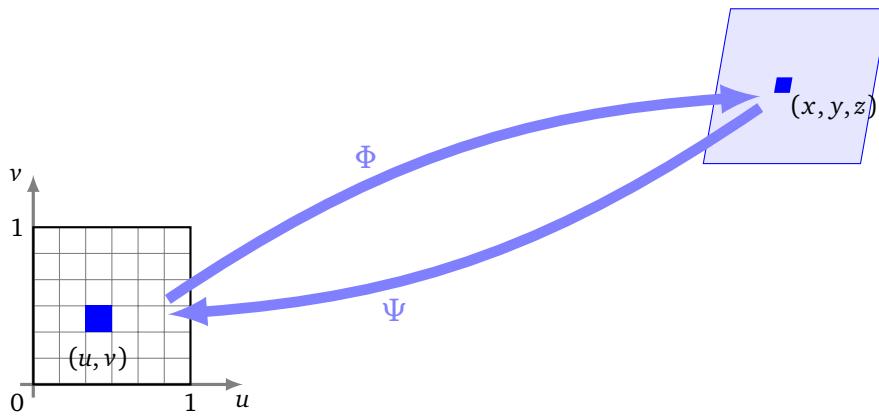
C'est une fonction qui à chaque point (u, v) d'un carré de référence associe une valeur. Autrement dit chaque pixel du carré est colorié. Pour un objet O (de l'espace) et une de ses faces F , la **fonction de plaquage** est une application bijective Φ :

$$\begin{aligned}\Phi : [0, 1]^2 &\longrightarrow F \\ (u, v) &\longmapsto \Phi(u, v)\end{aligned}$$

Sa réciproque étant :

$$\begin{aligned}\Psi : F &\longrightarrow [0, 1]^2 \\ P &\longmapsto \Psi(P)\end{aligned}$$

Appliquer la texture, c'est pour chaque $P \in F$ lui associer la couleur $\mathcal{C}(\Psi(P))$.



1.4. Texture (dimension 1) : suite

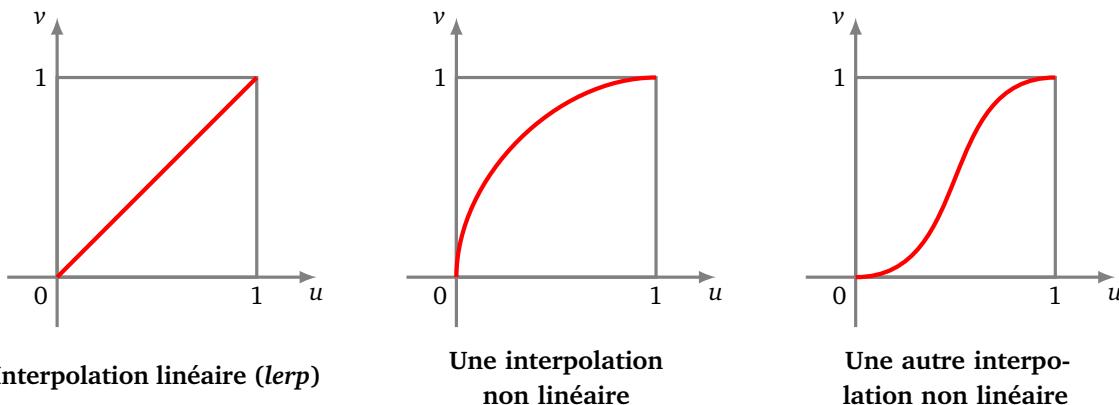
Revenons à nos textures en dimension 1.

Interpolation linéaire de $[0, 1]$ dans $[a, b]$. Tout d'abord, le choix de l'intervalle $[0, 1]$ comme ensemble de départ des fonctions \mathcal{C} et Φ est un peu arbitraire. On passe facilement de l'intervalle $[0, 1] \subset \mathbb{R}$ à n'importe quel intervalle $[a, b] \subset \mathbb{R}$ (avec $a \neq b$), pour cela il suffit de considérer la bijection :

$$\begin{aligned}\phi : [0, 1] &\longrightarrow [a, b] \\ u &\longmapsto (1-u)a + ub\end{aligned}$$

C'est l'exemple basique d'interpolation linéaire (*lerp*) : 0 s'envoie sur a , 1 sur b , $\frac{1}{2}$ sur $\frac{a+b}{2}$...

Interpolation non linéaire. Si $a = 0$ et $b = 1$ alors la fonction ϕ précédente est tout simplement l'identité $\phi(x) = x$. Mais il existe beaucoup d'autres fonctions $f : [0, 1] \rightarrow [0, 1]$ qui sont bijectives, par exemple toute fonction f continue, strictement croissante avec $f(0) = 0$ et $f(1) = 1$. Une analogie c'est parcourir 10 km en 1 h, on peut le faire à vitesse constante, ou bien rapide puis lent, ou l'inverse... De telles interpolations non linéaires peuvent être utiles.

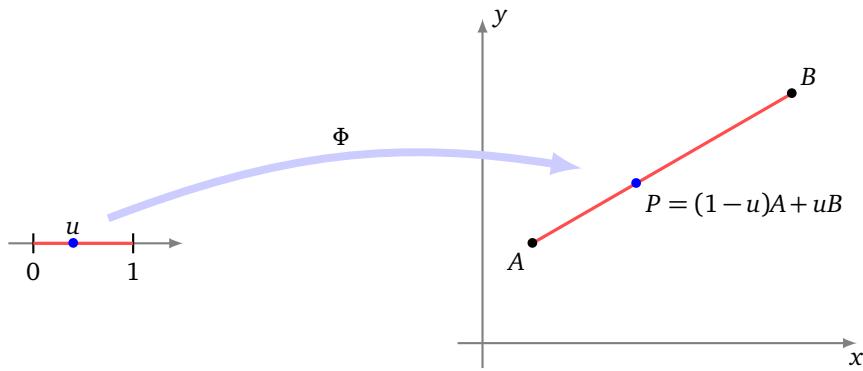


Dans la suite on se limitera aux interpolations linéaires.

Interpolation linéaire de $[0, 1]$ dans $[A, B]$.

Pour A, B deux points du plan (ou de l'espace), la fonction de plaquage $\Phi : [0, 1] \rightarrow [A, B]$ naturelle est celle de l'interpolation linéaire :

$$\Phi(u) = (1-u)A + uB.$$



Autrement dit, $P = (1-u)A + uB$, ou encore en termes de vecteurs $\vec{AP} = u\vec{AB}$. Si on note $A(x_A, y_A)$ et $B(x_B, y_B)$ alors :

$$P = \begin{pmatrix} x \\ y \end{pmatrix} = (1-u) \begin{pmatrix} x_A \\ y_A \end{pmatrix} + u \begin{pmatrix} x_B \\ y_B \end{pmatrix} = \begin{pmatrix} (1-u)x_A + ux_B \\ (1-u)y_A + uy_B \end{pmatrix} = \begin{pmatrix} x_A + u(x_B - x_A) \\ y_A + u(y_B - y_A) \end{pmatrix}.$$

Il s'agit en fait de l'interpolation linéaire sur chacune des coordonnées. Les formules sont similaires pour $A(x_A, y_A, z_A)$ et $B(x_B, y_B, z_B)$ des points de l'espace.

1.5. Équations directes et inverses

Interprétons les calculs précédents en termes de matrices. Les coordonnées (x, y) de $P = \Phi(u)$ se calculent par :

$$\begin{pmatrix} x \\ y \end{pmatrix} = T \begin{pmatrix} u \\ 1 \end{pmatrix}$$

avec

$$T = \begin{pmatrix} x_B - x_A & x_A \\ y_B - y_A & y_A \end{pmatrix}.$$

La preuve est juste une variante des écritures précédentes :

$$\begin{cases} x = u(x_B - x_A) + x_A \\ y = u(y_B - y_A) + y_A \end{cases}.$$

Ces formules définissent $\Phi : [0, 1] \rightarrow [A, B]$.

Réiproquement on pourrait calculer u à l'aide de

$$\begin{pmatrix} u \\ 1 \end{pmatrix} = T^{-1} \begin{pmatrix} x \\ y \end{pmatrix}$$

Mais un calcul direct donne ici :

$$u = \frac{x - x_A}{x_B - x_A} = \frac{y - y_A}{y_B - y_A}.$$

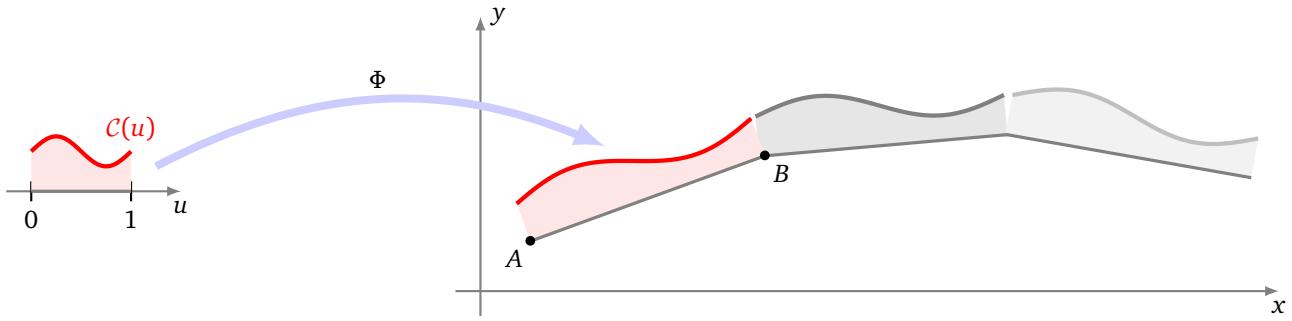
Cette formule définit $\Psi : [A, B] \rightarrow [0, 1]$.

1.6. Modifications selon la normale

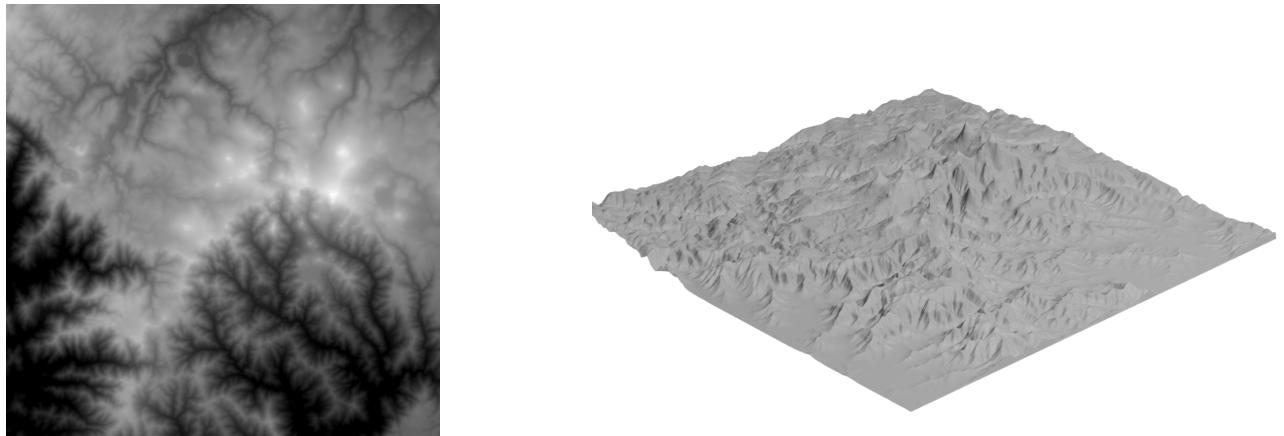
Les textures sont utiles bien au-delà de l'ajout d'une couleur. Les textures permettent : des effets de relief, de modifier la granulosité, modifier la matière, modifier l'effet de l'éclairage, d'ajouter de la transparence et de modifier localement la forme de l'objet.

Nous allons nous intéresser à la modification locale de la forme de l'objet qui peut être réelle ou simulée. Cela permet de donner du relief, comme par exemple la peau d'une orange. Nous allons voir trois méthodes de la plus simple à celle qui demande le plus de calculs.

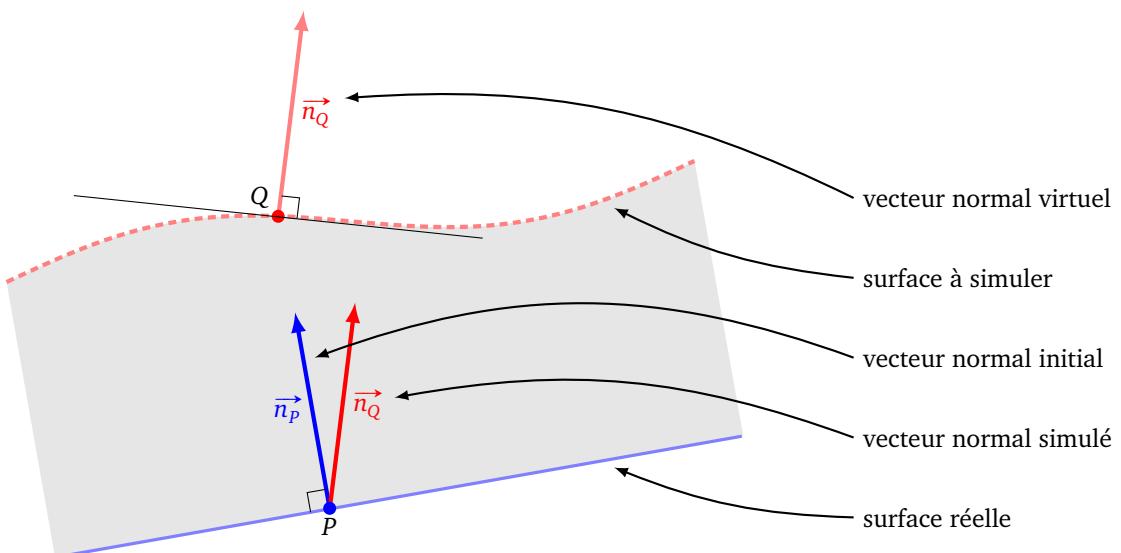
Bump map. Il s'agit de modifier la hauteur apparente de la surface en simulant un léger déplacement selon la direction normale. Mathématiquement, en dimension 1, cette texture est une fonction $\mathcal{C} : [0, 1] \rightarrow \mathbb{R}$ qui à un paramètre u associe une hauteur $\mathcal{C}(u)$ de déplacement.



En dimension 2 ce serait $\mathcal{C} : [0, 1]^2 \rightarrow \mathbb{R}$ et informatiquement, cette texture est stockée par une *heightmap*, c'est-à-dire une image en niveaux de gris qui représentent des hauteurs. Ci-dessous à gauche une image en niveaux de gris, les pixels blancs correspondant aux points les plus hauts et les pixels noirs aux points les plus bas. Sur la figure de droite le rendu 3D correspondant (ici un carré de côté 30 km autour de la source de la Loire), un facteur d'échelle permet d'obtenir un terrain plus ou moins plat (ici le rendu 3D est réel et correspond à la *displacement map* plus bas).



Revenons au cas d'une texture de dimension 1. Pour un point P de la face $[A, B]$ de paramètre $u = \Psi(P)$, au lieu d'afficher le point P on aimeraient afficher le point $Q = P + \mathcal{C}(u)\vec{n}_P$.

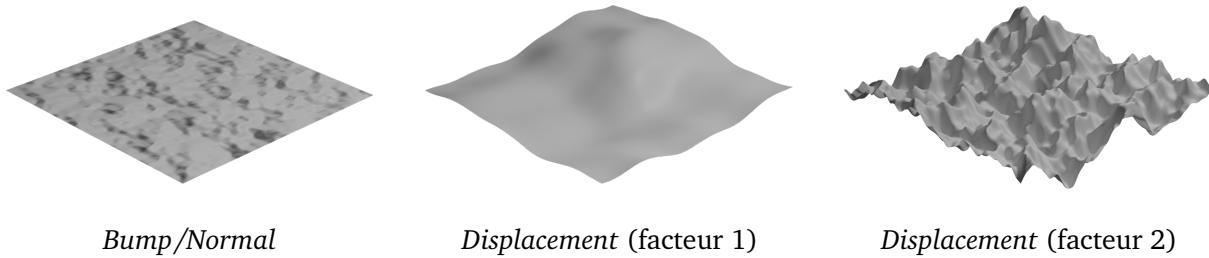


Cette modification est « simulée » car on ne va pas modifier la géométrie réelle de l'objet, on ne retient que la modification de la normale, autrement dit le changement d'éclairage de l'objet. Disons, pour simplifier,

que pour dessiner et éclairer un point d'un objet on a juste besoin de connaître la position de ce point P et le vecteur normal à l'objet en ce point \vec{n}_P (voir le chapitre « Lumière »). Ici on va bien tracer le point en P mais avec le vecteur normal \vec{n}_Q correspondant au point Q . Le calcul de \vec{n}_Q se fait à l'aide de \vec{n}_P et de la fonction $u \mapsto \mathcal{C}(u)$. Ces calculs sont simples et permettent de simuler efficacement des effets réalistes. Cependant la géométrie de l'objet n'a pas changé : un rayon lancé sur notre objet intersecte toujours l'objet sur la face initiale et pas sur la face simulée. Par exemple l'ombre de l'objet ne change pas.

Normal map. Il s'agit d'une variante de la modification précédente, mais cette fois la texture décrit le nouveau vecteur normal à considérer. Mathématiquement, en dimension 1, cette texture est une fonction $\mathcal{C} : [0, 1] \rightarrow \mathbb{R}^3$ qui à un paramètre u associe une modification du vecteur normal. En dimension 2 ce serait $\mathcal{C} : [0, 1]^2 \rightarrow \mathbb{R}^3$ et informatiquement, cette texture est stockée par une image couleur *rgb*, chaque niveau de rouge/vert/bleu désignant une des coordonnées (x, y, z) .

Il y a donc beaucoup plus de liberté que pour la *bump map* et l'effet peut être plus réaliste. Il permet même de réduire drastiquement la taille d'un maillage ou d'une triangulation tout en gardant un rendu 3D de qualité. Comme précédemment la géométrie réelle de l'objet n'est pas changée, on remplace le vecteur normal \vec{n}_P en P par un nouveau vecteur \vec{n}'_P calculé à partir de \vec{n}_P et de $\mathcal{C}(u)$. C'est ce nouveau vecteur \vec{n}'_P qui est utilisé en tant que vecteur normal lors de l'éclairage.

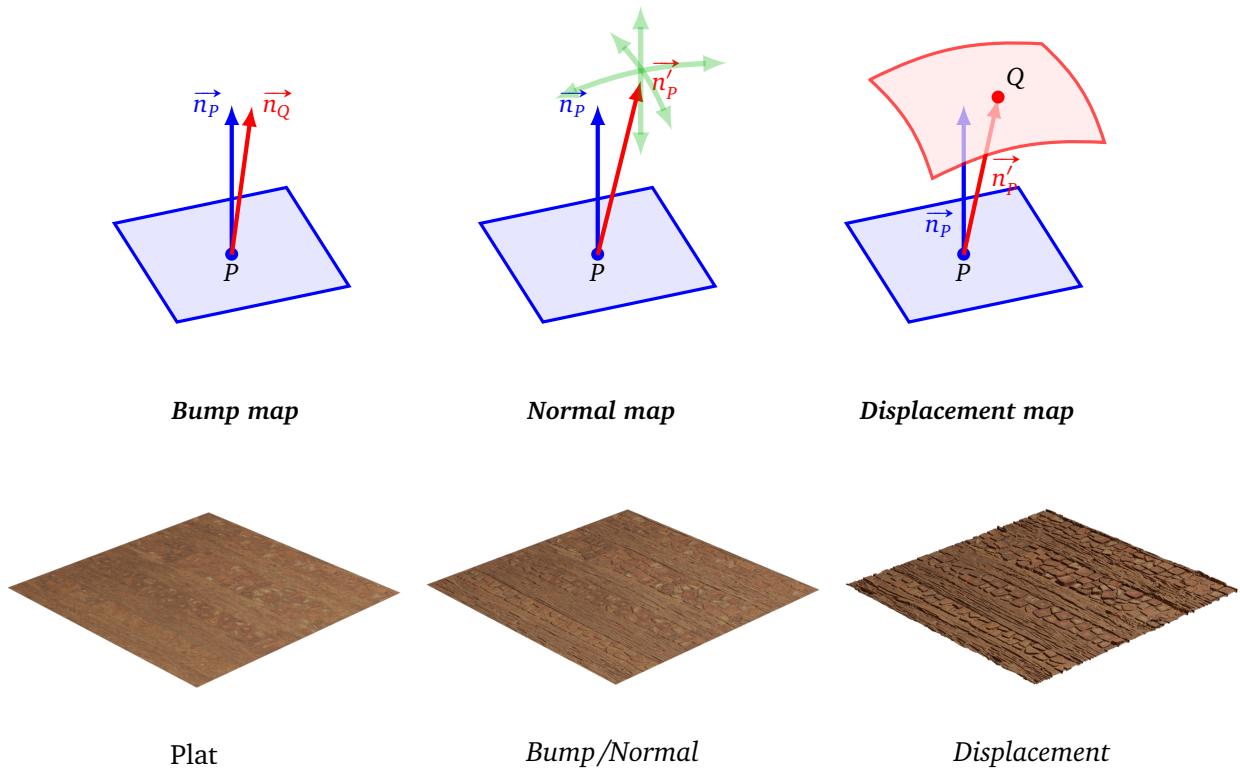


Displacement map. Cette modification est très similaire à la modification précédente, sauf que cette fois on déplace effectivement le point. La texture est de nouveau une application $\mathcal{C} : [0, 1] \rightarrow \mathbb{R}^3$ (dimension 1) ou $\mathcal{C} : [0, 1]^2 \rightarrow \mathbb{R}^3$ (dimension 2). Le point P est remplacé par le point $Q = P + \vec{n}'_P$. Ainsi tous les calculs pour les éclairages, les ombres, les lancers de rayons... se font à l'aide de cette nouvelle surface. Comme la surface déplacée est plus compliquée que la face originale les calculs sont plus longs.

Voici la déformation d'une sphère obtenue à partir d'un déplacement. Ici la texture est un bruit aléatoire. Un facteur d'échelle permet d'accentuer plus ou moins la déformation.



Ci-dessous un résumé des trois modifications : à gauche la *bump map*, la surface réelle de l'objet n'est pas modifiée mais le vecteur normal peut être changé avec un degré de liberté ; au centre la *normal map*, la surface réelle de l'objet n'est pas non plus modifiée mais le vecteur normal peut être changé avec trois degrés de liberté ; à droite la *displacement map* la surface réelle de l'objet est modifiée.



2. Texture en deux dimensions

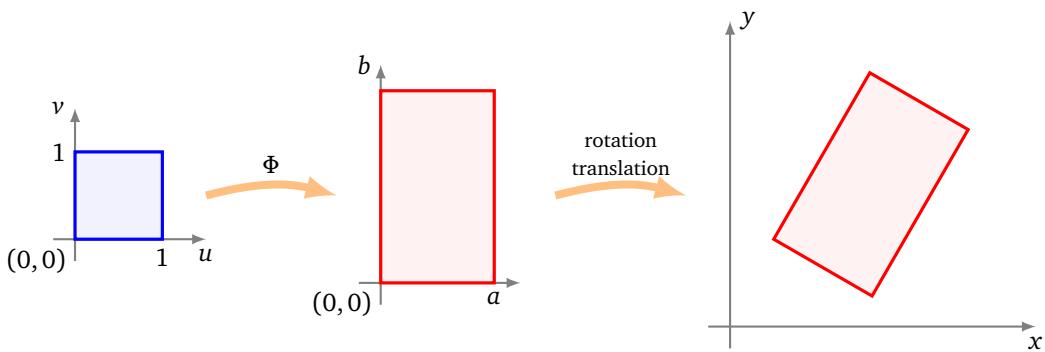
En partant d'une texture, comment l'appliquer sur l'objet réel ? Nous répondons à la question en fonction de la nature géométrique de la face des objets : triangle, rectangle, quadrilatère, cylindre ou sphère.

2.1. Rectangle sur rectangle/parallélogramme

Dans le plan, envoyer le carré unité sur un rectangle vertical est très simple. Par exemple $\Phi : (u, v) \mapsto (au, bv)$, envoie le carré $[0, 1] \times [0, 1]$ sur le rectangle $[0, a] \times [0, b]$. La matrice de cette application est :

$$M = \begin{pmatrix} a & 0 \\ 0 & b \end{pmatrix}.$$

Si on composait par une rotation puis une translation on pourrait obtenir n'importe quel rectangle du plan.



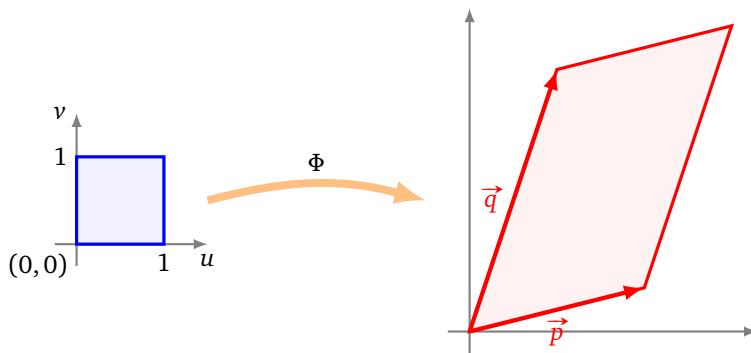
Plus généralement si $M = \begin{pmatrix} a & b \\ c & d \end{pmatrix} \in M_2(\mathbb{R})$ est une matrice inversible quelconque, la **transformation vectorielle** associée est

$$\Phi \begin{pmatrix} u \\ v \end{pmatrix} = M \begin{pmatrix} u \\ v \end{pmatrix}$$

c'est-à-dire $\Phi(u, v) = (x, y)$ où :

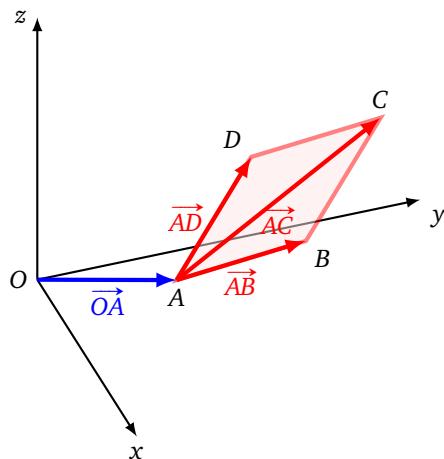
$$\begin{cases} x &= au + bv \\ y &= cu + dv \end{cases}.$$

Le vecteur $\vec{i} = \begin{pmatrix} 1 \\ 0 \end{pmatrix}$ de la base canonique s'envoie sur le vecteur $\vec{p} = \begin{pmatrix} a \\ c \end{pmatrix}$ et le vecteur $\vec{j} = \begin{pmatrix} 0 \\ 1 \end{pmatrix}$ s'envoie sur le vecteur $\vec{q} = \begin{pmatrix} b \\ d \end{pmatrix}$. Ainsi le carré unité s'envoie sur le parallélogramme engendré par (\vec{p}, \vec{q}) . N'importe quel parallélogramme du plan se construit ainsi comme l'image du carré unité par une transformation vectorielle (suivie d'une translation).



Dans l'espace, un parallélogramme $ABCD$ est défini par quatre points (distincts et coplanaires) tels que $\overrightarrow{AB} + \overrightarrow{AD} = \overrightarrow{AC}$. Si on note $\vec{p} = \overrightarrow{AB} = \begin{pmatrix} x_p \\ y_p \\ z_p \end{pmatrix}$ et $\vec{q} = \overrightarrow{AD} = \begin{pmatrix} x_q \\ y_q \\ z_q \end{pmatrix}$ alors la transformation de matrice $M \in M_{3,2}(\mathbb{R})$ (suivie d'une translation de vecteur \overrightarrow{OA}) envoie le carré unité du plan sur notre parallélogramme $ABCD$ de l'espace :

$$M = \begin{pmatrix} x_p & x_q \\ y_p & y_q \\ z_p & z_q \end{pmatrix}.$$

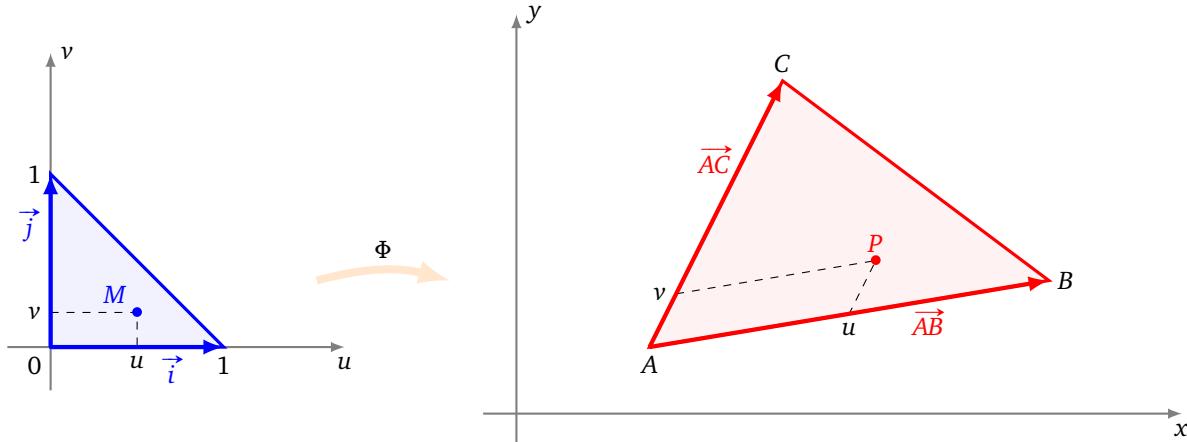


Nous verrons ultérieurement comment on peut envoyer le carré unité sur un quadrilatère quelconque. Noter cependant que 4 points dans l'espace ne sont en général pas dans un même plan, c'est pourquoi il est beaucoup plus naturel de considérer le cas de trois points de l'espace, car trois points de l'espace sont toujours coplanaires.

2.2. Coordonnées barycentriques

Triangle sur triangle.

Le but est d'envoyer le triangle « unité » du plan sur un triangle ABC quelconque du plan ou de l'espace. Cette transformation Φ doit envoyer \vec{i} sur \vec{AB} et \vec{j} sur \vec{AC} . Il existe une unique telle transformation affine Φ .

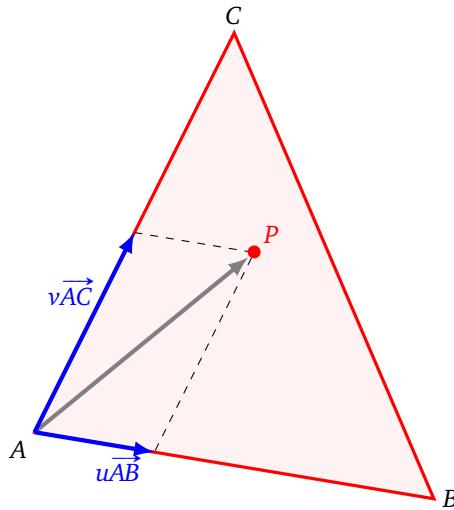


Considérons $M(u, v)$ un point du triangle unité de départ. Que (u, v) soient les coordonnées de M signifie $\overrightarrow{OM} = u \vec{i} + v \vec{j}$, c'est-à-dire $M = O + u \vec{i} + v \vec{j}$. Alors $P = \Phi(M)$ où :

$$\overrightarrow{AP} = u \overrightarrow{AB} + v \overrightarrow{AC}$$

c'est-à-dire

$$P = A + u \overrightarrow{AB} + v \overrightarrow{AC}$$

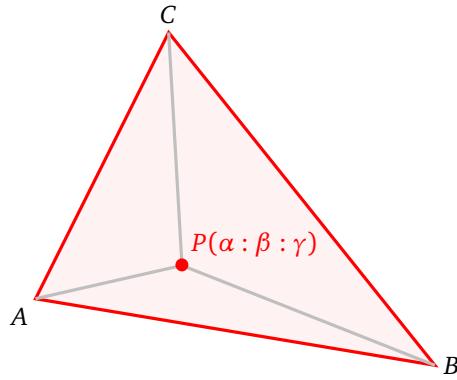


Noter que ces calculs sont valables aussi bien pour A, B, C, P points du plan ou de l'espace : dans tous les cas seulement deux coordonnées u, v sont nécessaires.

Barycentre. Soient A, B, C trois points du plan ou de l'espace. Le point P est **barycentre** de (A, α) , (B, β) , (C, γ) (avec des réels α, β, γ vérifiant $\alpha + \beta + \gamma \neq 0$) si :

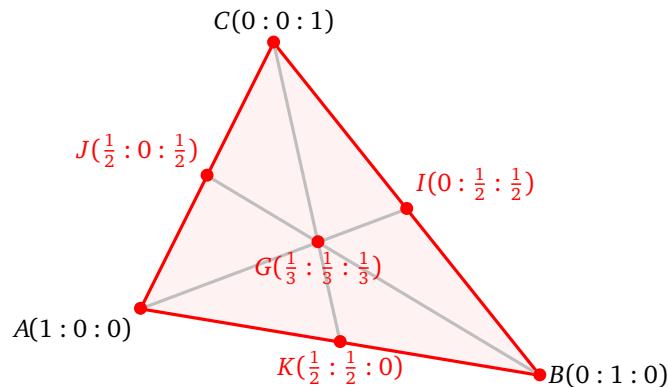
$$\alpha \overrightarrow{PA} + \beta \overrightarrow{PB} + \gamma \overrightarrow{PC} = \vec{0}.$$

On appelle alors $(\alpha : \beta : \gamma)$ les **coordonnées barycentriques** de P par rapport au triangle ABC . Les coordonnées sont dites **normalisées** si $\alpha + \beta + \gamma = 1$.



Si $(\alpha : \beta : \gamma)$ sont des coordonnées barycentriques, alors $(\lambda\alpha : \lambda\beta : \lambda\gamma)$ le sont aussi pour tout $\lambda \in \mathbb{R}^*$ (voir les coordonnées homogènes du chapitre « Transformations de l'espace »). Ainsi par multiplication par un facteur $\lambda = \frac{1}{\alpha+\beta+\gamma}$ on peut toujours se ramener à des coordonnées barycentriques normalisées.

Exemples.

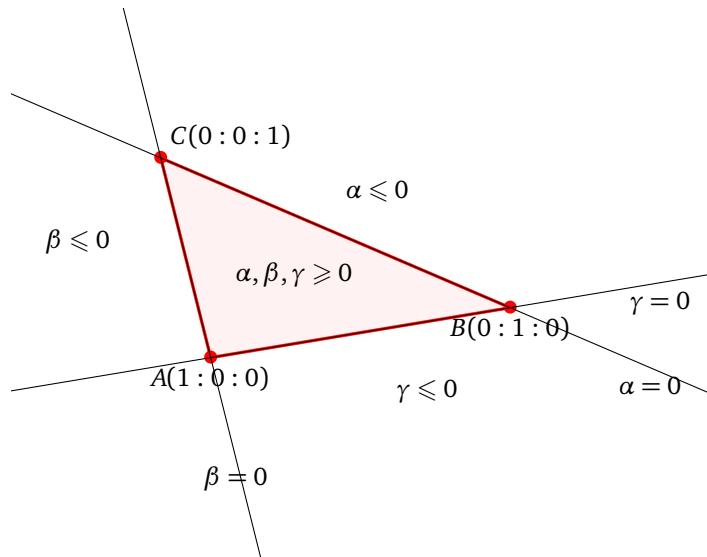


Les calculs avec les coordonnées barycentriques sont pratiques : le milieu de deux points $P(\alpha : \beta : \gamma)$ et $Q(\alpha' : \beta' : \gamma')$ est $\frac{P+Q}{2}$ de coordonnées barycentriques $(\frac{\alpha+\alpha'}{2} : \frac{\beta+\beta'}{2} : \frac{\gamma+\gamma'}{2})$.

L'**isobarycentre** G de A, B, C est $G = \frac{A+B+C}{3}$ et a pour coordonnées barycentriques $(\frac{1}{3} : \frac{1}{3} : \frac{1}{3}) = (1 : 1 : 1)$. Il vérifie $\vec{GA} + \vec{GB} + \vec{GC} = \vec{0}$.

Quelques remarques.

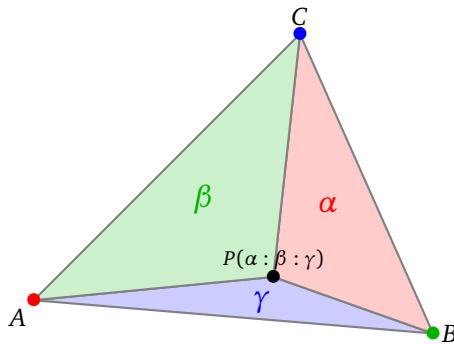
- Intuitivement, plus $P(\alpha : \beta : \gamma)$ est proche de A , plus α est grand ; plus P est proche de B , plus β est grand ; plus P est proche de C , plus γ est grand.
- P est situé dans le triangle ABC si et seulement si $\alpha \geq 0$, $\beta \geq 0$ et $\gamma \geq 0$.
- P est situé sur la droite (BC) si et seulement si $\alpha = 0$.



Aires.

On peut déterminer géométriquement les coordonnées barycentriques $(\alpha : \beta : \gamma)$ d'un point P . Chaque coefficient est égal à l'aire du triangle opposé divisée par l'aire du triangle total :

$$\alpha = \frac{\mathcal{A}_\Delta(PBC)}{\mathcal{A}_\Delta(ABC)} \quad \beta = \frac{\mathcal{A}_\Delta(PCA)}{\mathcal{A}_\Delta(ABC)} \quad \gamma = \frac{\mathcal{A}_\Delta(PAB)}{\mathcal{A}_\Delta(ABC)}$$



Ces formules sont valables pour un point P à l'intérieur du triangle. Nous verrons la preuve de ces formules un peu plus tard. On rappelle que l'on calcule facilement l'aire (sans signe) d'un triangle à l'aide des formules suivantes :

$$\mathcal{A}_\Delta(ABC) = \frac{1}{2} \|\vec{AB} \wedge \vec{AC}\|$$

Cette formule est valable dans le plan et l'espace. Dans le plan uniquement on a aussi :

$$\mathcal{A}_\Delta(ABC) = \frac{1}{2} |\det(\vec{AB}, \vec{AC})|$$

2.3. Conversions

Passage de (u, v) à $(\alpha : \beta : \gamma)$.

Il y a 3 coordonnées barycentriques α, β, γ , mais comme elles sont définies à un facteur multiplicatif λ près, il y a seulement deux degrés de liberté.

Proposition 1.

- Si $P = A + u\vec{AB} + v\vec{AC}$ alors P a pour coordonnées barycentriques $(1 - u - v : u : v)$.
- Réciproquement, si P a pour coordonnées barycentriques $(\alpha : \beta : \gamma)$ alors $P = A + u\vec{AB} + v\vec{AC}$ avec $u = \frac{\beta}{\alpha + \beta + \gamma}$ et $v = \frac{\gamma}{\alpha + \beta + \gamma}$.

Démonstration. La preuve c'est une reformulation de l'écriture $P = A + u\vec{AB} + v\vec{AC}$ sous la forme $\vec{PA} + u\vec{AB} + v\vec{AC} = \vec{0}$. On décompose $\vec{AB} = \vec{AP} + \vec{PB}$ et $\vec{AC} = \vec{AP} + \vec{PC}$ pour obtenir $(1-u-v)\vec{PA} + u\vec{PB} + v\vec{PC} = \vec{0}$. \square

Passage de (u, v) à (x, y) .

Nous avons deux façons de repérer un point P du plan ou de l'espace, d'une part les coordonnées cartésiennes (x, y) ou (x, y, z) et d'autre part les coordonnées (u, v) associées aux coordonnées barycentriques $(1-u-v : u : v)$. Voici comment passer de l'une à l'autre.

Proposition 2.

1. *Dans le plan*

$$\begin{pmatrix} x \\ y \end{pmatrix} = T_2 \begin{pmatrix} u \\ v \end{pmatrix} + \begin{pmatrix} x_A \\ y_A \end{pmatrix} \quad \begin{pmatrix} u \\ v \end{pmatrix} = T_2^{-1} \begin{pmatrix} x - x_A \\ y - y_A \end{pmatrix}$$

où

$$T_2 = \begin{pmatrix} x_B - x_A & x_C - x_A \\ y_B - y_A & y_C - y_A \end{pmatrix}.$$

2. *Dans l'espace*

$$\begin{pmatrix} x \\ y \\ z \end{pmatrix} = T_3 \begin{pmatrix} u \\ v \\ w \end{pmatrix} + \begin{pmatrix} x_A \\ y_A \\ z_A \end{pmatrix} \quad \text{où} \quad T_3 = \begin{pmatrix} x_B - x_A & x_C - x_A \\ y_B - y_A & y_C - y_A \\ z_B - z_A & z_C - z_A \end{pmatrix}.$$

On a

$$\begin{pmatrix} u \\ v \\ w \end{pmatrix} = T_2^{-1} \begin{pmatrix} x - x_A \\ y - y_A \\ z - z_A \end{pmatrix}$$

où T_2 est la matrice du cas précédent.

Démonstration. Il s'agit encore une fois juste de l'écriture de l'égalité $P = A + u\vec{AB} + v\vec{AC}$ en coordonnées cartésiennes. \square

On sait que si $M = \begin{pmatrix} a & b \\ c & d \end{pmatrix}$ est inversible alors

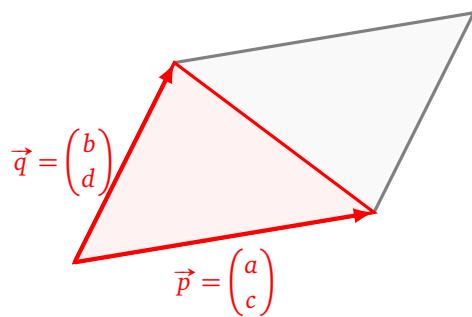
$$M^{-1} = \frac{1}{\det M} \begin{pmatrix} d & -b \\ c & a \end{pmatrix}$$

où

$$\det M = \begin{vmatrix} a & b \\ c & d \end{vmatrix} = ad - bc.$$

On rappelle que $\det(M)$ est l'aire (avec signe) du parallélogramme formé par $\vec{p} = \begin{pmatrix} a \\ c \end{pmatrix}$ et $\vec{q} = \begin{pmatrix} b \\ d \end{pmatrix}$.

Autrement dit c'est le double de l'aire du triangle. Ainsi $\mathcal{A}_\Delta = \frac{1}{2} \det(M)$.



Intéressons-nous au calcul : $\begin{pmatrix} u \\ v \end{pmatrix} = T_2^{-1} \begin{pmatrix} x - x_A \\ y - y_A \end{pmatrix}$. On obtient alors

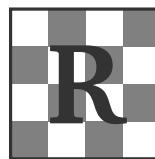
$$u = \frac{\begin{vmatrix} x - x_A & x_C - x_A \\ y - y_A & y_C - y_A \end{vmatrix}}{\begin{vmatrix} x_B - x_A & x_C - x_A \\ y_B - y_A & y_C - y_A \end{vmatrix}} = \frac{\mathcal{A}_\Delta(\overrightarrow{AP}, \overrightarrow{AC})}{\mathcal{A}_\Delta(\overrightarrow{AB}, \overrightarrow{AC})} \quad \text{et} \quad v = \frac{\begin{vmatrix} x_B - x_A & x - x_A \\ y_B - y_A & y - y_A \end{vmatrix}}{\begin{vmatrix} x_B - x_A & x_C - x_A \\ y_B - y_A & y_C - y_A \end{vmatrix}} = \frac{\mathcal{A}_\Delta(\overrightarrow{AB}, \overrightarrow{AP})}{\mathcal{A}_\Delta(\overrightarrow{AB}, \overrightarrow{AC})}$$

Ce sont les formules des coordonnées barycentriques calculées avec des aires de triangles comme on l'avait annoncé plus haut.

3. Autres figures géométriques

3.1. Répétitions

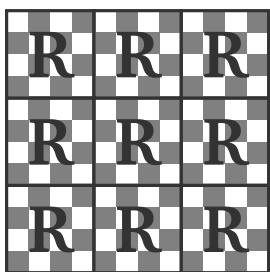
Si la face à décorer est grande, il peut être utile de réaliser la texture par répétitions d'un motif de base.



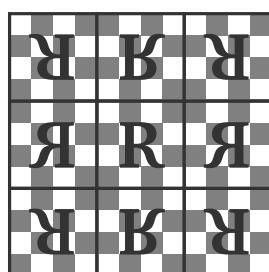
Motif de base

Voici plusieurs façons d'étendre un motif :

- *répétition* : le motif de base est répété par translations,
- *miroir* : le motif de base est répété par symétries axiales,
- *bord* : on entoure le motif d'une couleur neutre, pour éviter les problèmes au bord de la face lors de l'application de la texture,
- *extension/clamp* : on étend le motif en rajoutant une bande de sécurité tout autour du motif (cette bande n'est pas sensé servir au-delà d'une largeur d'un pixel), c'est encore une fois pour éviter les problèmes de bord.



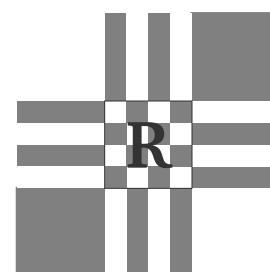
Répétition



Miroir



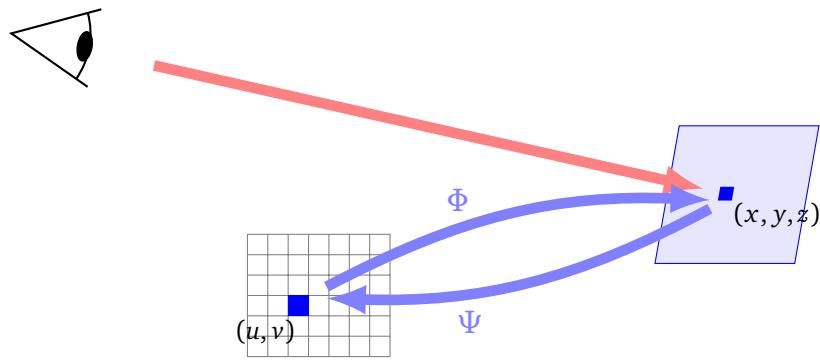
Bord



Extension

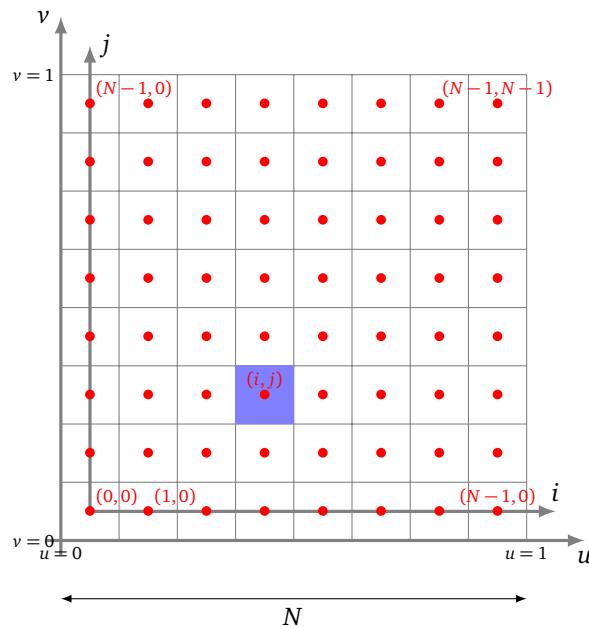
3.2. Pixel le plus proche

Jusqu'à présent nous avons présenté le texturage avec des calculs exacts où $(u, v) \in [0, 1]^2$ et $(x, y, z) \in \mathbb{R}^3$. Mais dans la pratique on travaille avec des coordonnées qui correspondent à des pixels. On rappelle comment se déroule le texturage inverse : pour colorier le point (x, y, z) d'un objet, on lui associe les coordonnées réelles $(u, v) \in [0, 1]^2$ et on le colorie selon la couleur $\mathcal{C}(u, v)$.



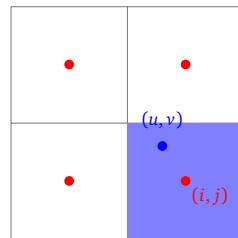
Mais comment concrètement décider ce qu'est $\mathcal{C}(u, v)$ à partir d'une image définissant la texture ?

Pixel le plus proche. Considérons une image carrée de taille $N \times N$ pixels, on note (i, j) avec $0 \leq i, j < N$ les coordonnées des centres des pixels. Identifions cette même image avec le carré unité $[0, 1]^2$ de coordonnées (u, v) . Noter que les origines de ces deux systèmes de coordonnées sont différentes.



Au centre (i, j) d'un pixel on associe les réels

$$(u, v) = \left(\frac{i + \frac{1}{2}}{N}, \frac{j + \frac{1}{2}}{N} \right) \in [0, 1]^2$$



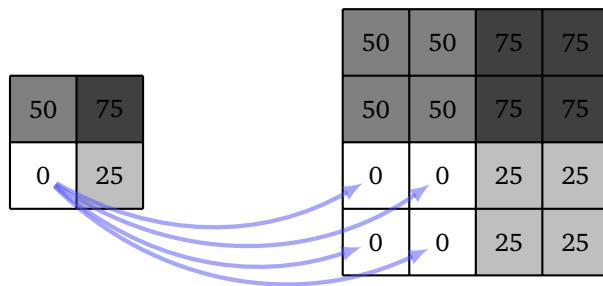
Dans l'autre sens, c'est plus délicat, puisque (u, v) ne correspond généralement pas au centre d'un pixel. On envoie donc (u, v) sur le centre du pixel le plus proche :

$$(i, j) = (\lfloor Nu \rfloor, \lfloor Nv \rfloor)$$

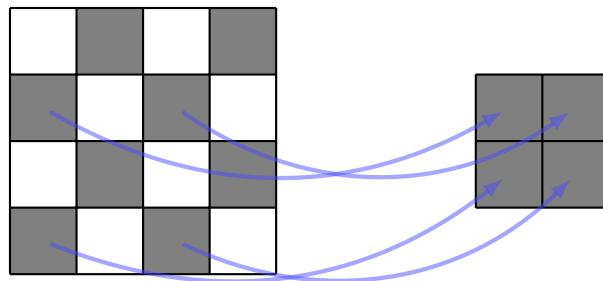
(sauf si $u = 1$ auquel cas $i = N - 1$ ou $v = 1$ auquel cas $j = N - 1$). On rappelle que $\lfloor x \rfloor$ désigne la partie entière de x .

Ce choix pose plusieurs problèmes.

D'une part si la texture est petite (elle a très peu de pixels) par rapport à la face de l'objet à colorier, alors le rendu sera de mauvaise qualité et fera apparaître une pixellisation.



D'autre part peuvent apparaître des problèmes d'échantillonage pour des grandes textures périodiques.



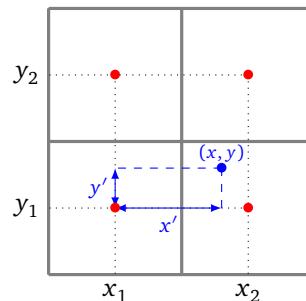
3.3. Interpolation bilinéaire

Pour éviter les problèmes précédents, il est raisonnable de tenir compte de la couleur de plusieurs pixels proches et pas seulement de celui le plus près. L'interpolation bilinéaire tient compte des 4 pixels les plus proches.

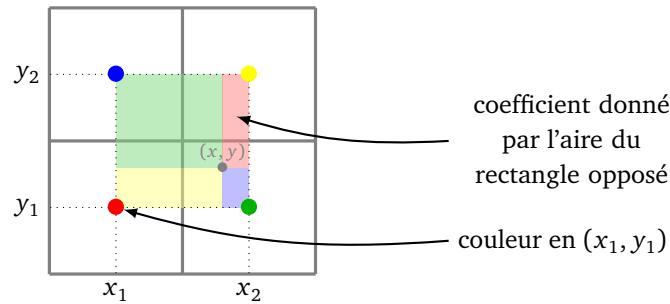
De façon générale considérons une fonction $F(x, y)$ qui pour l'instant est définie uniquement pour des valeurs entières, c'est-à-dire on connaît $F(x_i, y_j)$ pour $x_i, y_j \in \mathbb{Z}$. Comment interpoler une valeur pour $F(x, y)$ avec $(x, y) \in \mathbb{R}^2$?

On note $(x_1, y_1), (x_1, y_2), (x_2, y_1), (x_2, y_2)$ les sommets à coordonnées entières du carré qui entoure (x, y) . On note $x' = x - x_1$ et $y' = y - y_1$. Alors l'**interpolation bilinéaire** de F est définie par :

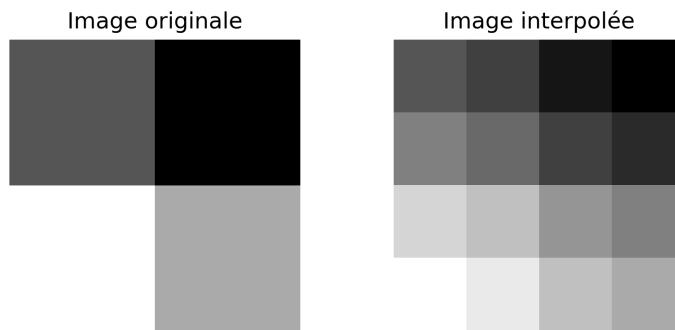
$$\begin{aligned} F(x, y) = & (1 - x')(1 - y')F(x_1, y_1) \\ & + (1 - x')y'F(x_1, y_2) \\ & + x'(1 - y')F(x_2, y_1) \\ & + x'y'F(x_2, y_2) \end{aligned}$$



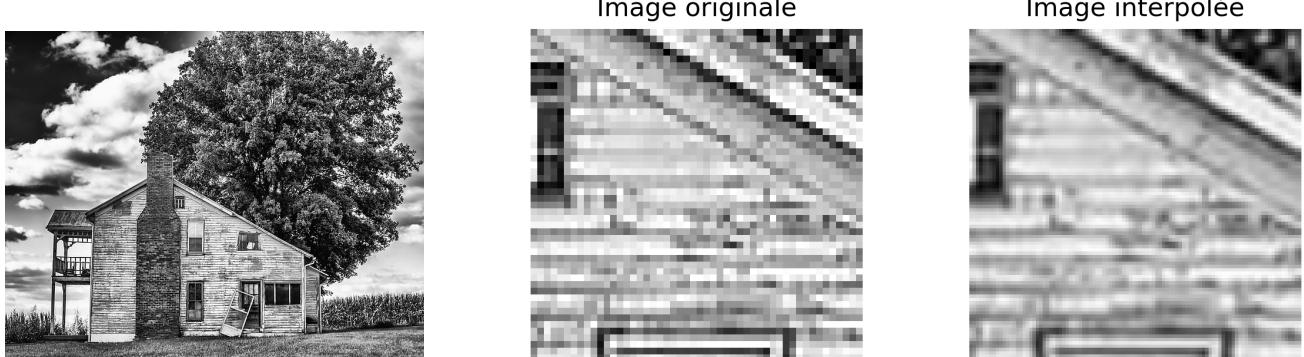
Le coefficient associé à $F(x_1, y_1)$ est l'aire du rectangle opposé à ce sommet. De même pour les autres coefficients. Ainsi plus (x, y) est proche d'un sommet, plus la valeur en ce sommet a d'importance.



Revenons à nos pixels, pour un point $(u, v) \in [0, 1]^2$ du carré unité, on lui associe les coordonnées réelles $(x, y) = (N(u + \frac{1}{2}), N(v + \frac{1}{2}))$ de l'image. On note $(x_1, y_1) = (\lfloor x \rfloor, \lfloor y \rfloor)$. Puis on pose $x_2 = x_1 + 1$ et $y_2 = y_1 + 1$. Reprenons notre exemple d'une image 2×2 (à gauche ci-dessous), l'interpolation bilinéaire permet de la transformer en une image 4×4 .



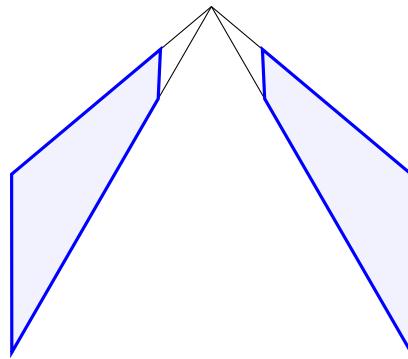
L'interpolation bilinéaire permet un agrandissement artificiel d'une image. Considérons une image (à gauche), on zoome sur une petite portion d'image qui apparaît alors pixelisée (au centre), l'interpolation bilinéaire permet d'obtenir une image plus lisse (à droite).



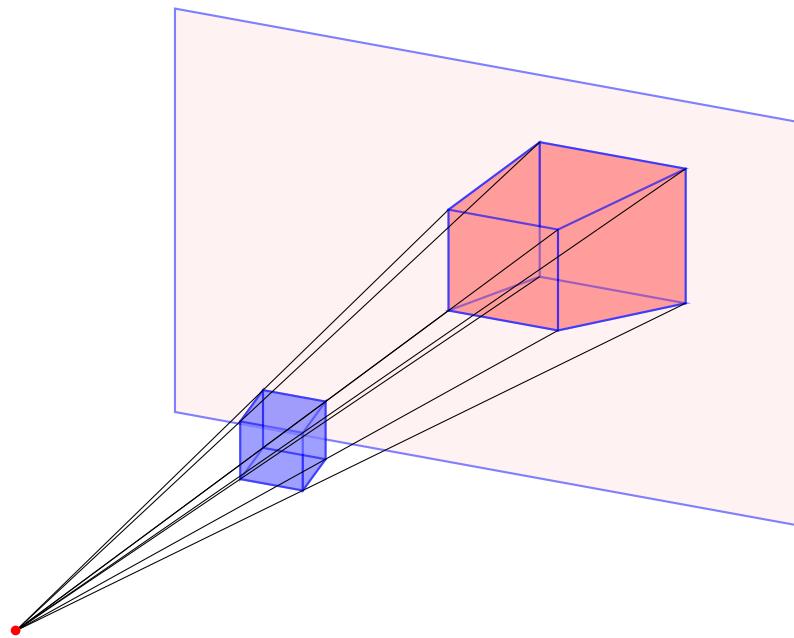
Bien sûr on pourrait faire mieux avec plus de pixels. Par exemple l'interpolation bicubique prend en compte une moyenne calculée sur les 16 pixels voisins. Mais les calculs sont longs et pas applicables à un rendu en temps réel.

3.4. Quadrilatère

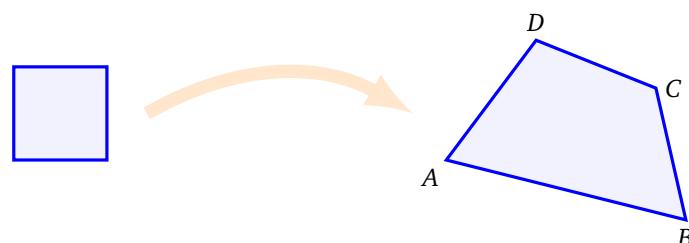
Le maillage des objets 3D est souvent réalisé par des triangulations. Mais il peut être plus adapté d'utiliser un maillage qui contient des quadrillatères. Un **quadrilatère** (*quad*) de l'espace, ce sont 4 points coplanaires (le quadrilatère est généralement supposé convexe).



Imaginez que vous êtes dans une rue entourée de deux murs, ces deux murs sont des rectangles de l'espace, mais ce que vous voyez en perspective ce sont des quadrillatères. Représenter ce mur est donc lié à la vision d'un objet (ici un rectangle) dans l'espace et a été traité en détail dans le chapitre « Perspective ». Cependant notre texture elle n'est pas un objet de l'espace. Il faut donc explicitement décrire l'application Φ qui permet de décorer ce mur (qui est un quadrilatère à l'écran).

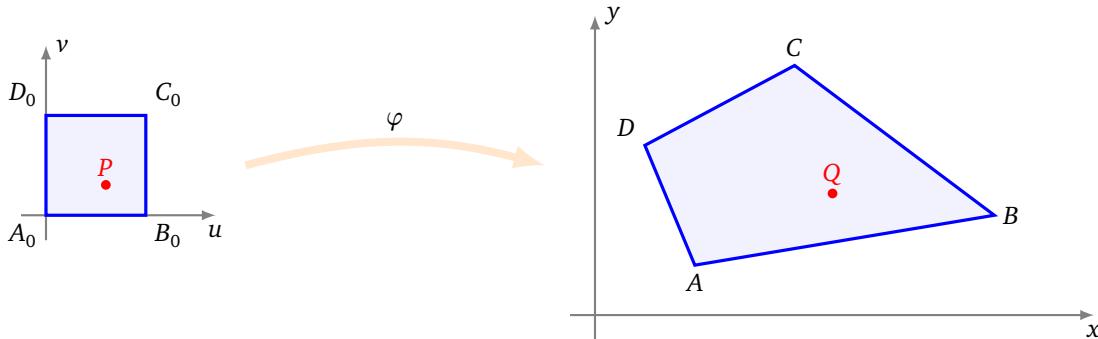


Le problème se ramène donc à envoyer notre carré unité $[0, 1]^2$ sur un quadrilatère convexe $ABCD$ quelconque du plan.

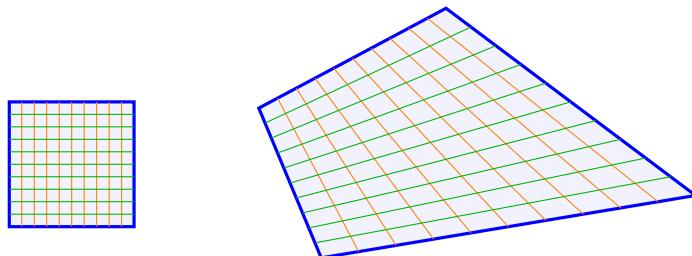


Interpolation bilinéaire. On considère le carré unité $A_0B_0C_0D_0$ de coordonnées $(u, v) \in [0, 1]^2$. Soit un quadrilatère convexe $ABCD$. L'interpolation bilinéaire qui envoie le carré sur le quadrilatère est :

$$\begin{aligned}\phi(u, v) = & (1-u)(1-v)A \\ & + u(1-v)B \\ & + uvC \\ & + (1-u)vD\end{aligned}$$



Dans ces calculs on identifie les points avec des vecteurs (ou bien leurs coordonnées) $A(x_A, y_A), \dots$ C'est l'interpolation bilinéaire précédente (en faisant bien attention aux coefficients associés à chaque sommet). Les calculs sont très faciles. Comme son nom l'indique cette fonction est linéaire dans les deux directions : un segment vertical du carré est envoyé sur un segment inclus dans le quadrilatère, un segment horizontal aussi. Par contre cette représentation du quadrilatère ne correspond pas exactement à la réalité d'un carré vu en perspective.



Transformation projective.

La fonction qui envoie un carré sur un quadrilatère et qui correspond à la perspective linéaire s'appelle une homographie du plan. Une **homographie** (ou **transformation projective**) est une application φ de \mathbb{R}^2 dans \mathbb{R}^2 définie par $\varphi : (u, v) \mapsto (x, y)$ où :

$$x = \frac{a_0 + a_1u + a_2v}{1 + c_1u + c_2v} \quad y = \frac{b_0 + b_1u + b_2v}{1 + c_1u + c_2v}$$

avec 8 coefficients réels $a_0, a_1, a_2, b_0, b_1, b_2, c_1$ et c_2 . (Noter que les deux dénominateurs de x et y sont identiques.)

Notons le carré unité $[0, 1]^2$ par $A_0B_0C_0D_0$ (en fait cela pourrait être un quadrilatère quelconque). Soient A, B, C, D quatre points du plan (formant aussi un quadrilatère). Alors il existe une unique homographie φ telle que :

$$\varphi(A_0) = A \quad \varphi(B_0) = B \quad \varphi(C_0) = C \quad \varphi(D_0) = D.$$

Comment déterminer cette homographie ? Il faut déterminer les 8 coefficients, pour cela nous avons 4 conditions : $\varphi(0, 0) = (x_A, y_A)$, $\varphi(1, 0) = (x_B, y_B)$, $\varphi(0, 1) = (x_D, y_D)$, $\varphi(1, 1) = (x_C, y_C)$. Chacune de ces conditions fournit deux équations. Par exemple $\varphi(1, 0) = (x_B, y_B)$ s'écrit (en posant $u = 1, v = 0$) : $x_B = \frac{a_0 + a_1}{1 + c_1}$ et $y_B = \frac{b_0 + b_1}{1 + c_1}$. En multipliant par le dénominateur on obtient 8 équations linéaires du type

$$(1 + c_1u + c_2v)x = a_0 + a_1u + a_2v \quad \text{ou} \quad (1 + c_1u + c_2v)y = b_0 + b_1u + b_2v$$

où les inconnues sont les 8 coefficients a_0, a_1, \dots, c_2 et les données sont les x parcourant $\{x_A, x_B, x_C, x_D\}$ et y parcourant $\{y_A, y_B, y_C, y_D\}$.

Nous obtenons donc un système de 8 équations linéaires avec 8 inconnues, qui admet une unique solution (sauf exceptions que l'on ne discutera pas ici). Résoudre ce système revient à inverser une matrice 8×8 (avec beaucoup de 0) ce qui est facile pour un ordinateur. Notons :

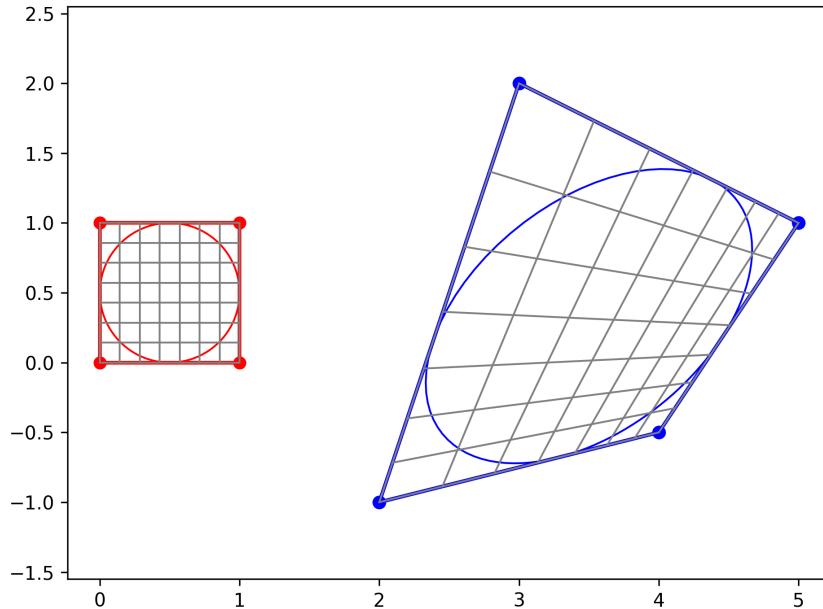
$$M = \begin{pmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 1 & 1 & 0 & 0 & 0 & 0 & -x_B & 0 \\ 0 & 0 & 0 & 1 & 1 & 0 & -y_B & 0 \\ 1 & 1 & 1 & 0 & 0 & 0 & -x_C & -x_C \\ 0 & 0 & 0 & 1 & 1 & 1 & -y_C & -y_C \\ 1 & 0 & 1 & 0 & 0 & 0 & 0 & -x_D \\ 0 & 0 & 0 & 1 & 0 & 1 & 0 & -y_D \end{pmatrix} \quad H = \begin{pmatrix} a_0 \\ a_1 \\ a_2 \\ b_0 \\ b_1 \\ b_2 \\ c_1 \\ c_2 \end{pmatrix} \quad Q = \begin{pmatrix} x_A \\ y_A \\ x_B \\ y_B \\ x_C \\ y_C \\ x_D \\ y_D \end{pmatrix}$$

Le système d'équations est $MH = Q$, ainsi on obtient les coefficients H de l'homographie en fonction des coordonnées Q du quadrilatère par :

$$H = M^{-1}Q.$$

Exemple. Sur la figure ci-dessous les sommets du quadrilatère $ABCD$ ont pour coordonnées : $(2, -1), (4, -\frac{1}{2}), (5, 1), (3, 2)$, qui forment le vecteur $Q = (2, -1, 4, -\frac{1}{2}, 5, 1, 3, 2)$. On trouve $H = (a_0, a_1, a_2, b_0, b_1, b_2, c_1, c_2) = (2, 5, -0.125, -1, 0.125, 2.25, 0.75, -0.375)$ et donc les équations définissant φ . On peut voir le quadrilatère bleu comme un carré vu en perspective.

Le calcul



Une façon plus efficace pour les calculs serait d'utiliser les coordonnées homogènes (voir le chapitre « Transformations de l'espace »). On rappelle qu'un point $(x, y) \in \mathbb{R}^2$ correspond aux coordonnées homogènes $(x : y : 1)$ et réciproquement $(x : y : z) = (\frac{x}{z} : \frac{y}{z} : 1)$ correspond au point $(\frac{x}{z}, \frac{y}{z}) \in \mathbb{R}^2$. La matrice d'une homographie (en coordonnées homogènes) est :

$$\bar{H} = \begin{pmatrix} a_1 & a_2 & a_0 \\ b_1 & b_2 & b_0 \\ c_1 & c_2 & 1 \end{pmatrix}$$

Il est facile de voir que :

$$\varphi(u, v) = \bar{H} \begin{pmatrix} u \\ v \\ 1 \end{pmatrix} = \begin{pmatrix} a_0 + a_1 u + a_2 v \\ b_0 + b_1 u + b_2 v \\ 1 + c_1 u + c_2 v \end{pmatrix}$$

Ce qui en coordonnées homogènes est exactement $(x : y : 1)$ et correspond bien à (x, y) . Il existe des méthodes pour déterminer les coefficients de \bar{H} sans résoudre le système linéaire 8×8 , mais elles sont plus sophistiquées et nous ne les détaillerons pas ici.

L'application inverse $\psi = \varphi^{-1}$ transforme le quadrilatère en un carré. C'est très utile pour « redresser » une image qui n'a pas été prise avec le bon angle de vue. Ci-dessous à gauche l'image originale, à droite sa transformation projective bien choisie.



Voici les équations de $\psi = \varphi^{-1} : (x, y) \mapsto (u, v) :$

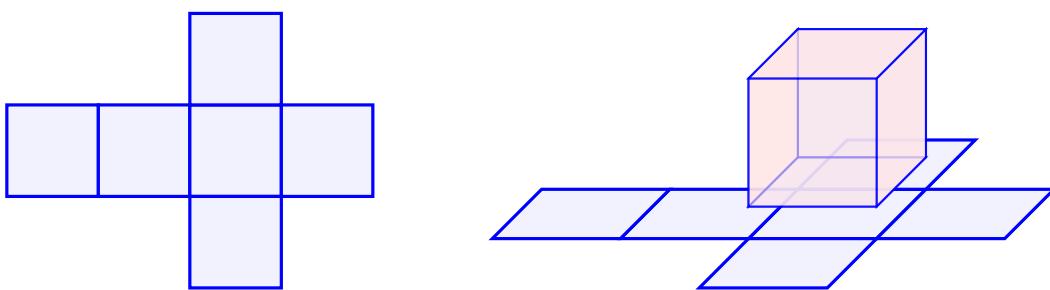
$$u = \frac{-a_0 b_2 + a_2 b_0 + (-b_0 c_2 + b_2)x + (a_0 c_2 - a_2)y}{a_1 b_2 - a_2 b_1 + (b_1 c_2 - b_2 c_1)x + (-a_1 c_2 + a_2 c_1)y}$$

et

$$v = -\frac{-a_0 b_1 + a_1 b_0 + (-b_0 c_1 + b_1)x + (a_0 c_1 - a_1)y}{a_1 b_2 - a_2 b_1 + (b_1 c_2 - b_2 c_1)x + (-a_1 c_2 + a_2 c_1)y}.$$

3.5. Cube

Pour texturer un cube, on se donne souvent l'image du patron regroupant les 6 faces dépliées.



3.6. Cylindre

Pour appliquer une texture sur un cylindre ou une portion de ce cylindre on utilise bien sûr les coordonnées cylindriques (voir le chapitre « Trigonométrie »). Par exemple pour appliquer une texture de $[0, 1]^2$ vers un cylindre de rayon r et de hauteur h , on applique la fonction :

$$\Phi : (u, v) \longmapsto (r, \theta, z) = (r, 2\pi u, hv)$$

où (r, θ, z) sont les coordonnées cylindriques.



Si on souhaite n'appliquer la texture que sur une petite partie du cylindre (par exemple une petite étiquette sur un bouteille) et si cette partie est délimitée en coordonnées cylindriques par $\theta_1 \leq \theta \leq \theta_2$ et $z_1 \leq z \leq z_2$ alors l'application de texture serait :

$$\Phi : (u, v) \mapsto (r, \theta, z) = \left(r, (1-u)\theta_1 + u\theta_2, (1-v)z_1 + vz_2 \right).$$

3.7. Sphère

Pour appliquer une texture sur une sphère de rayon r on utilise les coordonnées sphériques (r, φ, λ) (où $\varphi \in [-\frac{\pi}{2}, \frac{\pi}{2}]$ est la latitude et $\lambda \in [-\pi, \pi]$) est la longitude) :

$$\Phi : (u, v) \mapsto (r, \varphi, \lambda) = \left(r, \pi(v - \frac{1}{2}), 2\pi(u - \frac{1}{2}) \right).$$

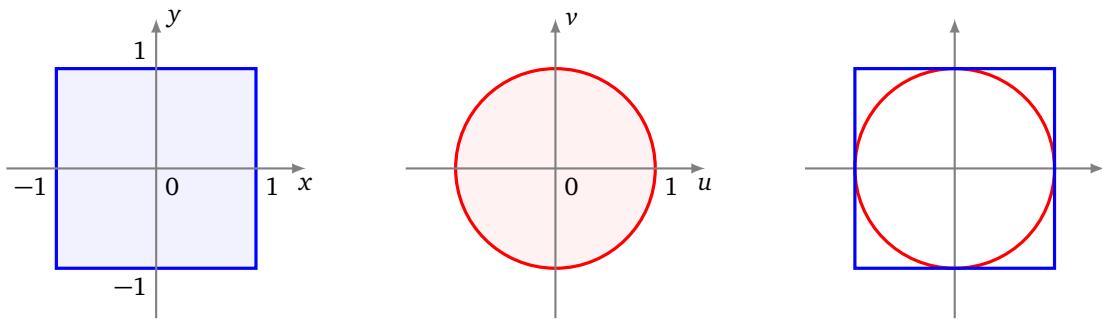
Le centre $(\frac{1}{2}, \frac{1}{2})$ de la texture est envoyé sur le point de la sphère de latitude et longitude nulle.



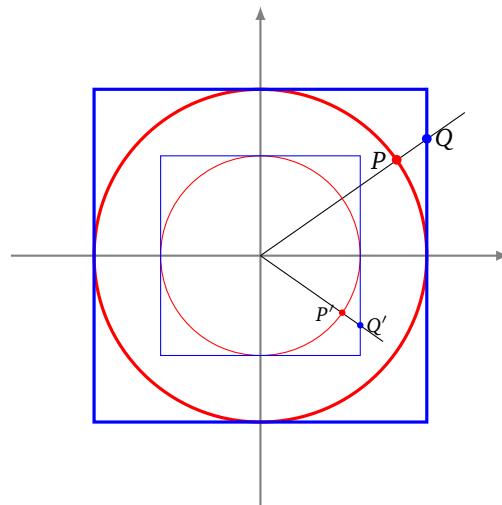
3.8. Du cube vers la sphère

Nous avons vu qu'il est très facile de réaliser le texturage d'un cube à l'aide de son patron. Pour texturer une sphère il peut être intéressant de d'abord texturer un cube puis de le déformer en une sphère. Comme les équations générales peuvent devenir assez compliquées on va s'intéresser au problème du plan : comment transformer un carré vers un cercle (et inversement) ?

Considérons le carré $C = [-1, 1]^2$ (de côté de longueur 2) et le disque (fermé) D de centre $(0, 0)$ et de rayon 1. On note (x, y) avec $-1 \leq x \leq 1$ et $-1 \leq y \leq 1$ les coordonnées pour le carré et (u, v) avec $u^2 + v^2 \leq 1$ les coordonnées pour le disque. Il y a plusieurs façons de déformer un disque en un carré



Méthode 1 : rayons. Le disque D est inclus dans le carré C . Pour un point P du bord du disque D , on trace le rayon issu de l'origine, ce rayon recoupe le bord du carré C en Q qui sera l'image de P . Si maintenant P est un point intérieur à D , il est situé sur un cercle de rayon r centré à l'origine, on effectue le même procédé pour envoyer ce point sur le carré de côté $2r$. Ainsi chaque cercle concentrique est envoyé sur un carré.



Les équations $F : D \rightarrow C$, $(u, v) \mapsto (x, y)$ pour passer d'un disque à un carré sont données par :

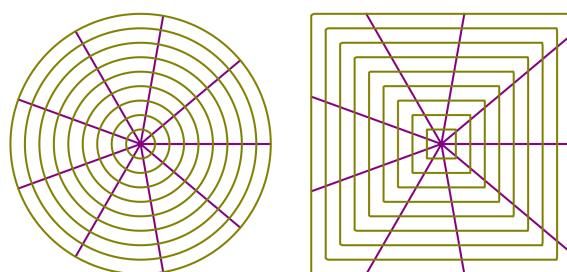
$$x = \begin{cases} \operatorname{sgn}(u)\sqrt{u^2 + v^2} & \text{si } u^2 \geq v^2 \\ \operatorname{sgn}(v)\frac{u}{v}\sqrt{u^2 + v^2} & \text{si } u^2 < v^2 \end{cases} \quad y = \begin{cases} \operatorname{sgn}(u)\frac{v}{u}\sqrt{u^2 + v^2} & \text{si } u^2 \geq v^2 \\ \operatorname{sgn}(v)\sqrt{u^2 + v^2} & \text{si } u^2 < v^2 \end{cases}$$

La fonction signe, notée $\operatorname{sgn}(x)$, est définie par :

$$\operatorname{sgn}(x) = \begin{cases} +1 & \text{si } x > 0 \\ 0 & \text{si } x = 0 \\ -1 & \text{si } x < 0 \end{cases}$$

Pour $x \neq 0$ on a aussi $\operatorname{sgn}(x) = \frac{x}{|x|}$.

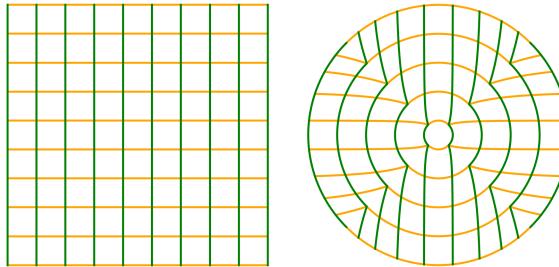
Voici le disque et sa transformation en un carré :



Les équations inverses $G : C \rightarrow D, (x, y) \mapsto (u, v)$ permettent de passer du carré au disque :

$$u = \begin{cases} \operatorname{sgn}(x) \frac{x^2}{\sqrt{x^2+y^2}} & \text{si } x^2 \geq y^2 \\ \operatorname{sgn}(y) \frac{xy}{\sqrt{x^2+y^2}} & \text{si } x^2 < y^2 \end{cases} \quad v = \begin{cases} \operatorname{sgn}(x) \frac{xy}{\sqrt{x^2+y^2}} & \text{si } x^2 \geq y^2 \\ \operatorname{sgn}(y) \frac{y^2}{\sqrt{x^2+y^2}} & \text{si } x^2 < y^2 \end{cases}$$

Voici le carré et sa transformation en un disque :



Méthode 2 : via des ellipses. Voyons une méthode un peu plus régulière pour déformer un disque en un carré. Un segment vertical du carré est envoyé sur une portion d'ellipse dans le disque, de même pour des segments horizontaux. Les équations $G : C \rightarrow D, (x, y) \mapsto (u, v)$ qui permettent de passer du carré au disque sont particulièrement simples :

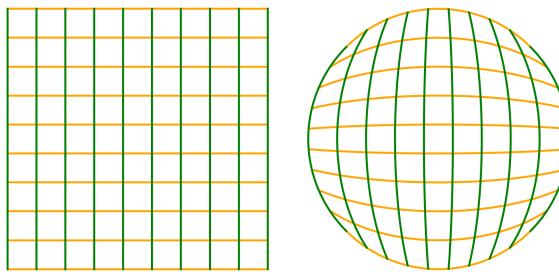
$$u = x \sqrt{1 - \frac{y^2}{2}} \quad v = y \sqrt{1 - \frac{x^2}{2}}$$

Les équations $F : D \rightarrow C, (u, v) \mapsto (x, y)$ pour passer d'un disque à un carré sont données par :

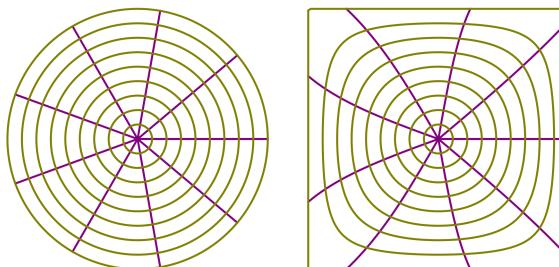
$$x = \frac{1}{2} \sqrt{2 + u^2 - v^2 + 2\sqrt{2}u} - \frac{1}{2} \sqrt{2 + u^2 - v^2 - 2\sqrt{2}u}$$

$$y = \frac{1}{2} \sqrt{2 - u^2 + v^2 + 2\sqrt{2}v} - \frac{1}{2} \sqrt{2 - u^2 + v^2 - 2\sqrt{2}v}$$

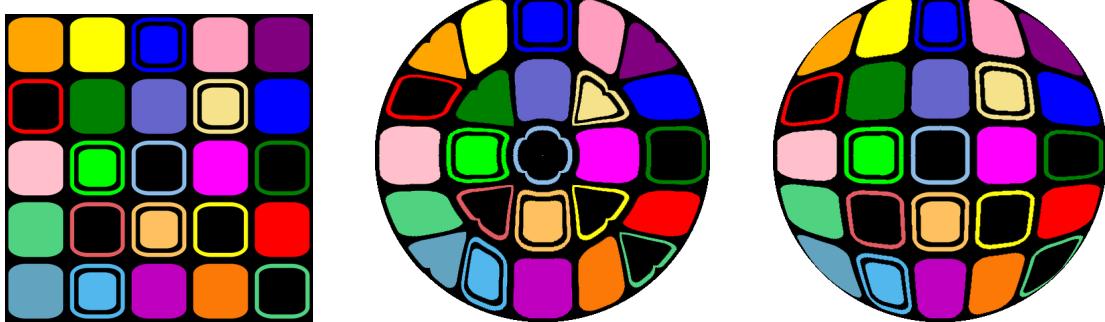
Voici le carré et sa transformation en un disque :



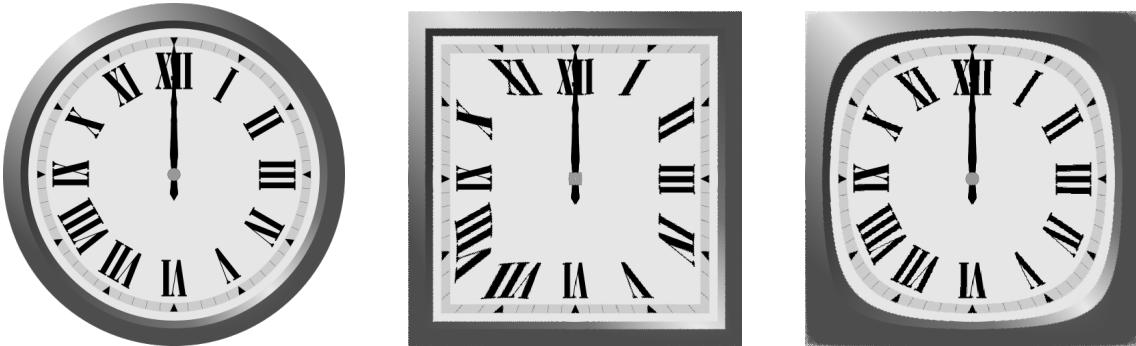
Voici le disque et sa transformation en un carré :



On transforme ainsi facilement une image carrée en une image ronde (à gauche l'image originale, au centre la méthode 1, à droite la méthode 2) :



Et une image ronde vers une image carrée (à gauche l'image originale, au centre la méthode 1, à droite la méthode 2) :



Ces équations et beaucoup d'autres sont à retrouver dans [Analytical methods for squaring the disc](#) de Chamberlain Fong.

Du cube vers la sphère. Voici les équations de la fonction $G : (x, y, z) \mapsto (u, v, w)$, analogue à la version elliptique ci-dessus, qui permettent de passer du cube $[-1, 1]^3$ (avec les coordonnées (x, y, z)) à la boule de rayon 1 centrée à l'origine (définie par $u^2 + v^2 + w^2 \leqslant 1$).

$$\begin{aligned} u &= x \sqrt{1 - \frac{y^2}{2} - \frac{z^2}{2} + \frac{y^2 z^2}{3}} \\ v &= y \sqrt{1 - \frac{x^2}{2} - \frac{z^2}{2} + \frac{x^2 z^2}{3}} \\ w &= z \sqrt{1 - \frac{x^2}{2} - \frac{y^2}{2} + \frac{x^2 y^2}{3}} \end{aligned}$$

Lancer de rayons II

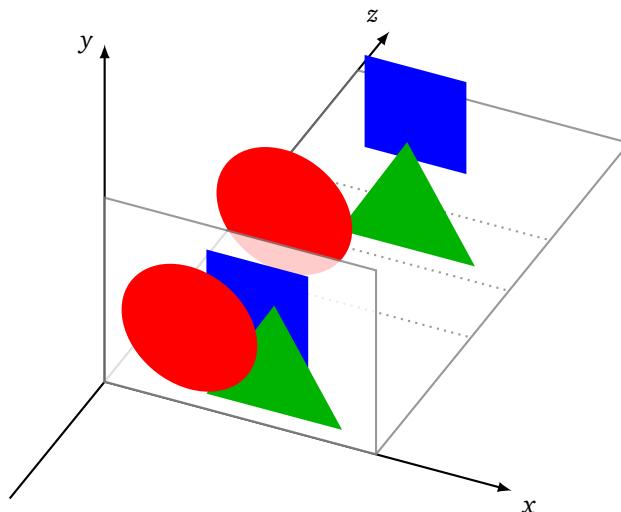
Nous expliquons en détail la technique du ray-tracing et présentons des outils pour accélérer cette méthode.

1. Les ancêtres du lancer de rayons

Nous allons voir deux algorithmes qui permettent de transformer une scène en une image comme si on voyait les objets du dessus et d'assez loin.

1.1. Algorithme du z -buffer

Il s'agit en quelque sorte de lancer des rayons parallèles à l'axe (Oz) et, pour chaque pixel, de calculer quel est l'objet le plus proche.



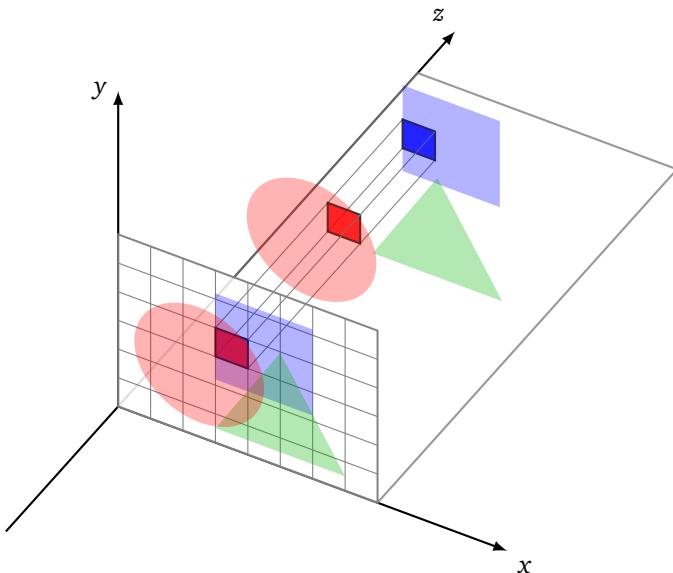
Nous avons des objets \mathcal{O}_k , chaque objet est un ensemble de (i, j, z, Cl) où z est la profondeur associée au pixel (i, j) et Cl est sa couleur (z et Cl dépendent de (i, j)). (Sur le dessin ci-dessus les objets de la scène sont en 2D mais pourraient aussi être en 3D.) L'écran ou l'image est un ensemble de (i, j, Cl) où Cl représente la couleur du pixel (i, j) . D'un point de vue matériel c'est une zone de la mémoire graphique (*buffer*). On initialise la couleur de tous les pixels à la couleur de fond (par exemple blanc ou transparent).

On a besoin en plus d'un *z-buffer* composé de (i, j, z) qui à la fin stocke la profondeur z de l'objet le plus proche pour les coordonnées (i, j) . On initialisera toutes les profondeurs à $+\infty$.

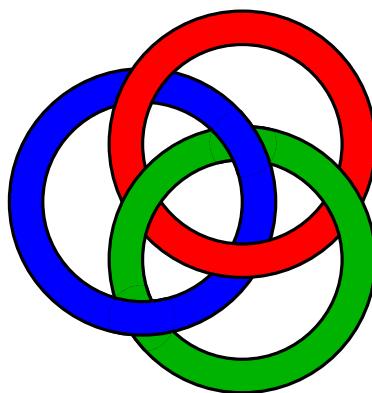
Algorithme (Algorithme du z -buffer).

Pour chaque (i, j) :

- Initialiser prof à $+\infty$.
- Pour chaque objet \mathcal{O}_k :
 - si (i, j, z, Cl) vérifie $z < \text{prof}$:
 - colorer le pixel (i, j) de l'écran avec la couleur Cl ,
 - faire $\text{prof} \leftarrow z$.

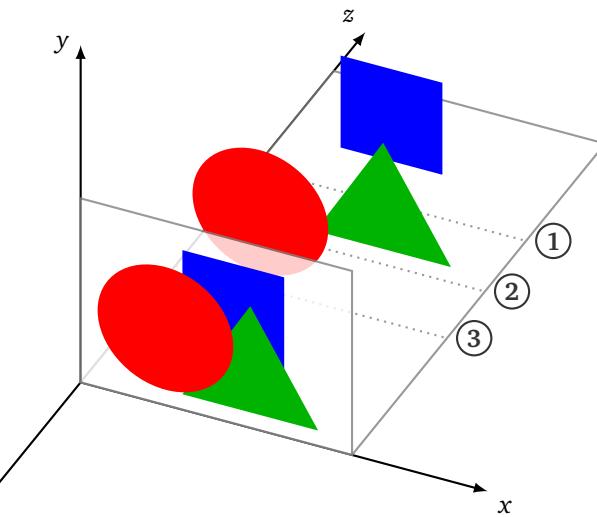


L'algorithme a le mérite de la simplicité, chaque pixel se voit attribuer la couleur de l'objet le plus proche. Il fonctionne même avec des objets entrelacés. La position des objets de la scène est rendue fidèlement du point de vue de l'observateur mais l'affichage ne tient pas compte de l'éclairage.

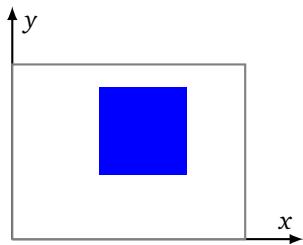
**1.2. Algorithme du peintre (z -sorting)**

Son nom vient de l'analogie avec un peintre qui part d'une toile blanche puis applique de la peinture en plusieurs touches et retouches ; à la fin seules sont visibles les couleurs de la couche supérieure. La démarche est semblable à la méthode précédente, mais ici on suppose que les objets ne s'entrelacent pas dans la profondeur.

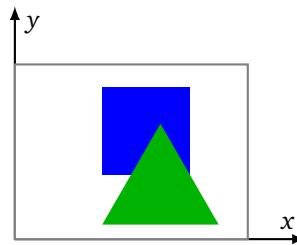
La première partie de l'algorithme consiste à ordonner les objets du plus loin (z grand) au plus proche (z petit).



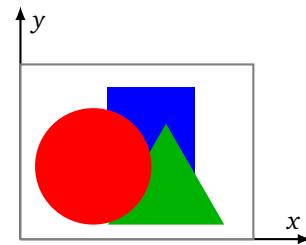
On dessine les objets un par un à l'écran (autrement dit on met à jour la partie de la mémoire de l'écran correspondant aux pixels de l'objet) en partant du plus éloigné et en terminant par le plus proche.



Étape ①



Étape ②



Étape ③

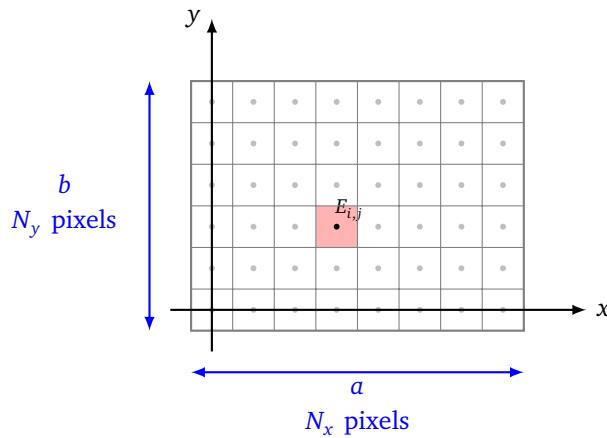
Un avantage de cette méthode est qu'il n'y a pas de test de profondeur à chaque pixel, on peut aussi facilement ajouter de la transparence aux objets. Parmi les inconvénients : on traite tous les objets, même ceux qui à la fin seront cachés ; il n'y a pas d'entrelacements possibles ; le classement préalable des objets n'est pas une tâche simple. Enfin on n'a pas résolu les problèmes soulevés avec l'algorithme du z -buffer : l'affichage ne tient pas compte de l'éclairage.

1.3. Écran

Expliquons comment lancer un rayon depuis un œil à travers un écran. Tout d'abord pour l'écran (ou l'image à afficher) nous allons fixer des coordonnées (x, y) . Supposons que les dimensions réelles de l'écran/l'image soit $a \times b$ et qu'en pixels la dimension soit $N_x \times N_y$. La largeur d'un pixel est alors a/N_x , sa hauteur b/N_y et on supposera $a/N_x = b/N_y$ pour avoir des pixels carrés. Le pixel indexé (i, j) sera donc considéré comme un carré centré aux coordonnées E_{ij} avec :

$$E_{ij} = \left(i \frac{a}{N_x}, j \frac{b}{N_y} \right)$$

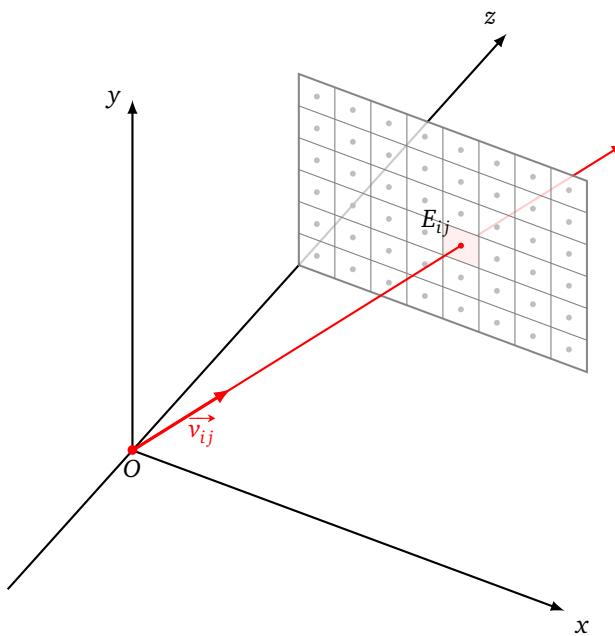
pour i entier variant de 0 à $N_x - 1$ et j entier variant de 0 à $N_y - 1$.



On peut rapidement passer d'un pixel à son voisin de droite en ajoutant $\frac{a}{N_x}$ (à calculer au préalable) à l'abscisse. Le pixel juste au-dessus s'obtient en ajoutant $\frac{b}{N_y}$ à l'ordonnée.

Revenons à notre scène 3D. Le repère a pour coordonnées (x, y, z) . L'œil qui sera l'origine des rayons est situé en $O(0, 0, 0)$. Nous plaçons l'écran dans le plan $(z = f)$ de sorte que les coordonnées du pixel (i, j) (que l'on note abusivement encore E_{ij}) soient $E_{ij} = \left(i \frac{a}{N_x}, j \frac{b}{N_y}, f\right)$.

Le rayon issu de O traversant le pixel (i, j) est donc porté par le vecteur $\vec{v}_{ij} = \overrightarrow{OE_{ij}}$. Si on préfère travailler avec des vecteurs unitaires on choisit $\vec{v}_{ij} = \frac{\overrightarrow{OE_{ij}}}{\| \overrightarrow{OE_{ij}} \|}$.



2. Principe du ray-tracing

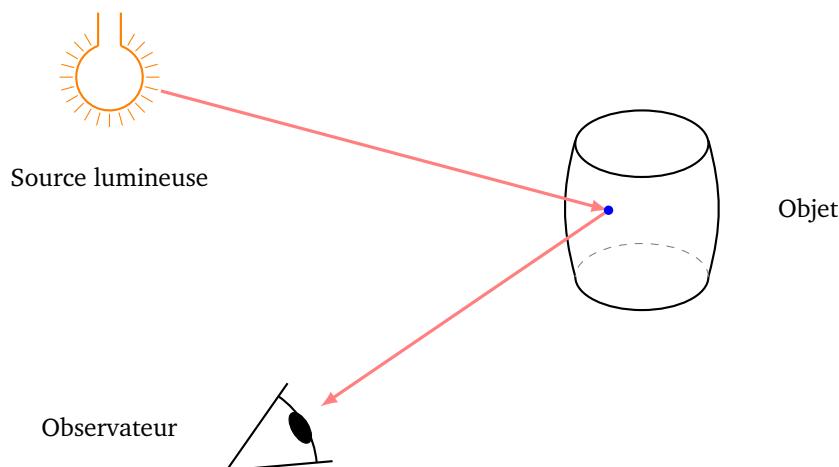
Motivations :

- nous avons déjà vu comment calculer l'intersection d'un rayon avec une surface élémentaire dans le chapitre « Lancer de rayons I »,
- nous avons aussi vu comment éclairer des objets dans le chapitre « Lumière »,
- il reste à expliquer comment afficher une scène à l'écran avec la méthode du *ray-tracing* qui fournit une modélisation réaliste des objets et de leur éclairage à partir des principes de la physique.

2.1. Lancer depuis la source lumineuse

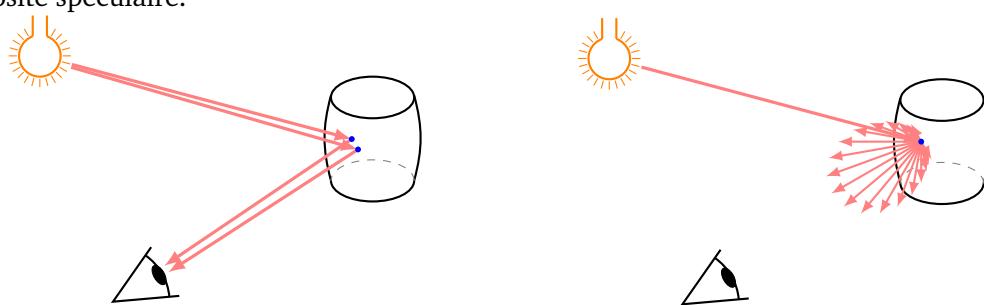
L'idée naturelle, et qui correspond à la réalité physique, est la suivante, dans sa version la plus simple :

- un rayon part de la source lumineuse,
- il vient frapper un objet,
- il y est réfléchi,
- et ce rayon réfléchi atteint la rétine (ou l'objectif de la caméra).



Les complications arrivent assez vite :

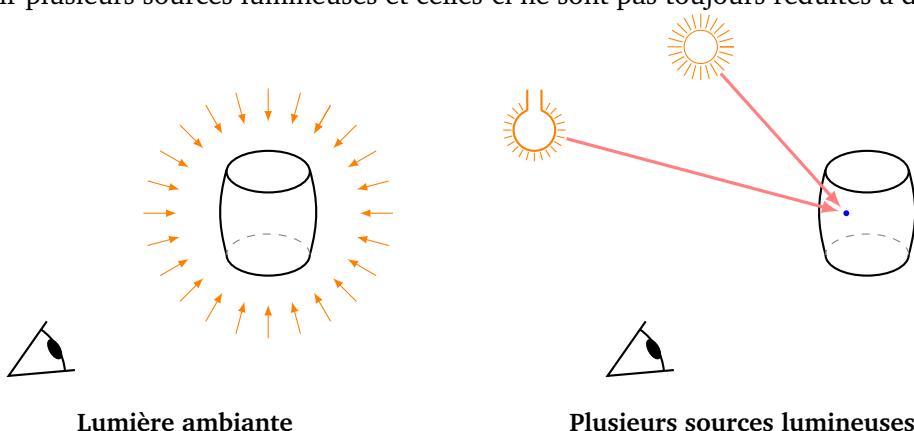
- La source lumineuse émet plusieurs rayons, éventuellement dans plusieurs directions, et donc plusieurs rayons atteignent la rétine.
- Un rayon ne se réfléchit pas toujours comme sur un miroir, mais rebondit vers différentes directions : on renvoie de nouveau au chapitre « Lumière » pour la composante de luminosité diffuse et la composante de luminosité spéculaire.



Plusieurs rayons arrivent à la rétine

Diffusion

- Il faut tenir compte de la lumière ambiante, qui provient de toutes les directions et éclaire partout, comme en extérieur avec un ciel couvert ou en intérieur avec des fenêtres voilées.
- Il peut y avoir plusieurs sources lumineuses et celles-ci ne sont pas toujours réduites à des points.



Lumière ambiante

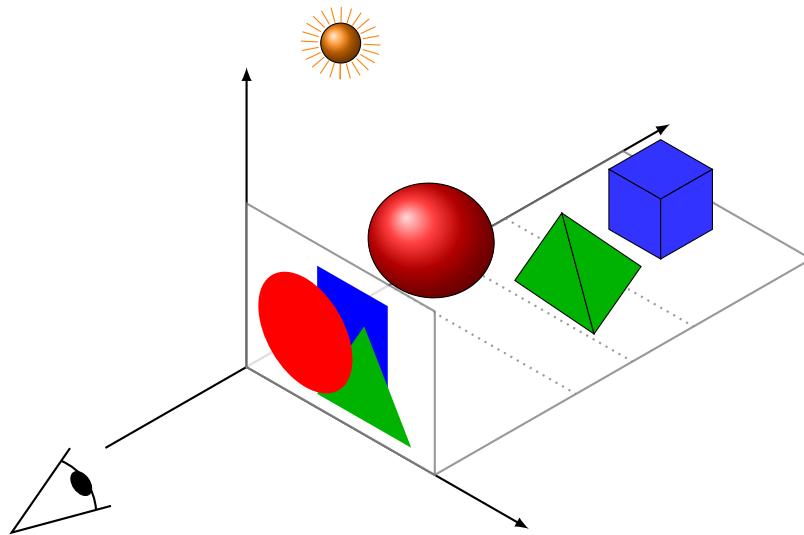
Plusieurs sources lumineuses

Au final il y a beaucoup de rayons ! Une infime partie seulement de ces rayons va atteindre notre rétine. C'est donc un énorme gaspillage de calculer le trajet de tous ces rayons en partant des sources lumineuses. Pensez au Soleil qui éclaire la moitié de la Terre, mais vous ne percevez qu'une minuscule partie de ses rayons.

2.2. Lancer depuis l'observateur

Nous savons que la rétine et l'objectif ne sont pas réduits à un point : ce sont des surfaces. Modélisons la situation :

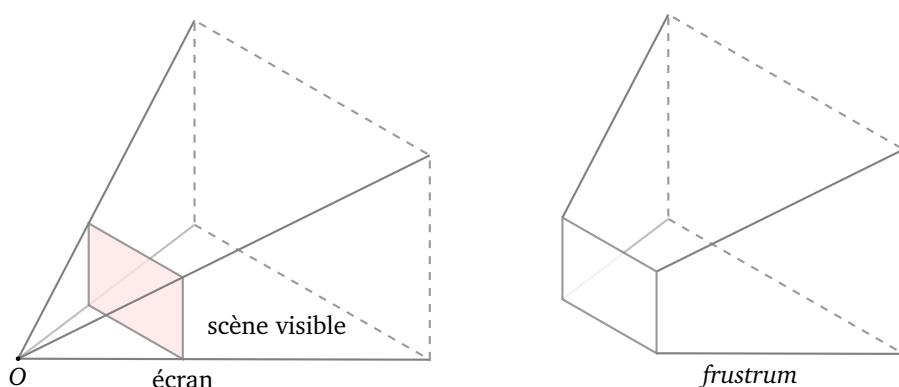
- le fond de l'œil est représenté par un point O ,
- la rétine ou l'objectif est modélisé par un écran (virtuel) qui est une grille de pixels,
- nous avons une scène composée d'objets 3D,
- et une source lumineuse.



L'objectif est de calculer l'image de la scène perçue depuis le point O à travers l'écran.

Idée générale : on lance un rayon depuis l'œil, à travers l'écran, jusqu'à intersester un objet de la scène, on remonte ensuite le trajet de la lumière.

On rencontre le mot *frustrum* pour l'ensemble de l'espace visible par l'œil à travers l'écran. D'un point de vue mathématique, le *frustrum* est un prisme tronqué à base rectangulaire. Ainsi on s'assure de ne calculer que des rayons visibles. Sur le dessin ci-dessous le prisme est tronqué au niveau de l'écran, mais s'étend vers l'infini dans la direction du rectangle pointillé.

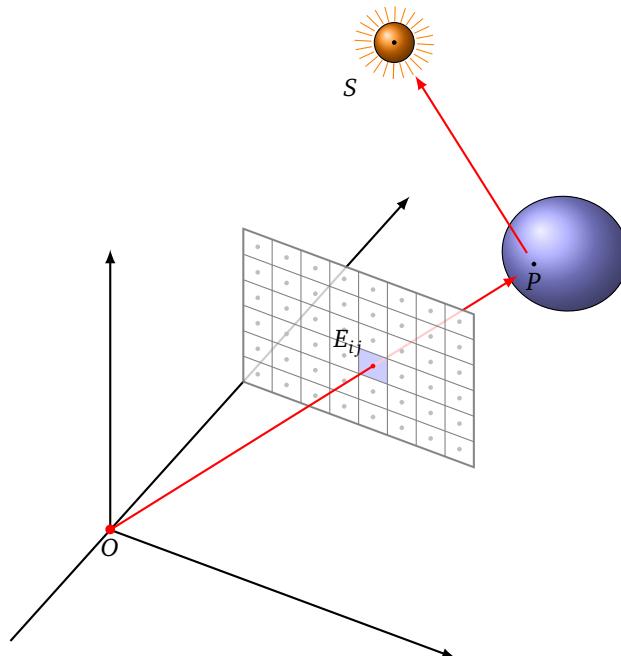


Voici le principe du *ray-tracing* dans sa version simple afin de colorier un pixel E_{ij} :

- On lance un rayon issu de l'œil O , passant par le (centre du) pixel E_{ij} .
- On calcule le premier point d'intersection P de ce rayon avec les objets de la scène.

- Depuis P on calcule la lumière reçue depuis la source lumineuse en lançant un rayon de P vers cette source S . On en déduit la couleur $Cl_{P \leftarrow O}$ de l'objet en P (perçue depuis le point O).
- On colorie le pixel E_{ij} selon cette couleur Cl . Si le rayon issu de O n'intersecte aucun objet, on colorie le pixel E_{ij} avec la couleur de fond (noir par exemple).

On itère ces étapes pour chaque pixel E_{ij} de l'écran.



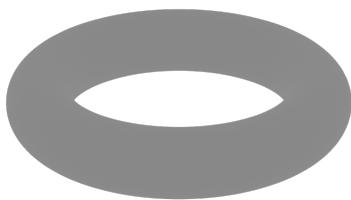
2.3. Rappel sur la lumière

Voici quelques rappels du chapitre « Lumière ». Le but est de calculer la couleur perçue d'un objet. Cette couleur se calcule dans le modèle de Phong par l'addition de trois types de luminosité :

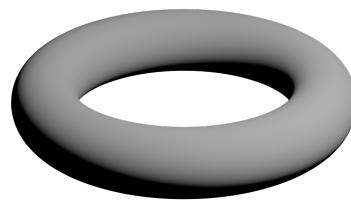
- luminosité ambiante Cl_{amb} ,
- luminosité diffuse Cl_{diff} ,
- luminosité spéculaire Cl_{spec} .

La couleur perçue est l'addition de ces couleurs :

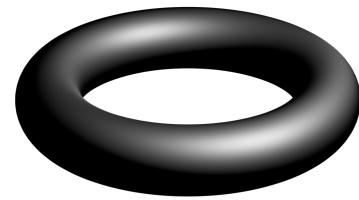
$$Cl_{\text{perçue}} = Cl_{\text{amb}} \oplus_{cl} Cl_{\text{diff}} \oplus_{cl} Cl_{\text{spec}}$$



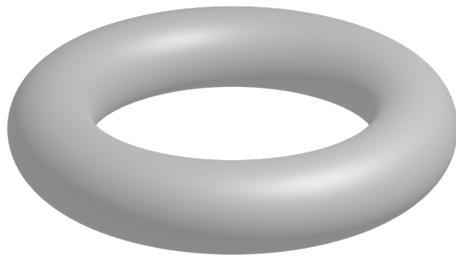
Lumière ambiante



Lumière diffuse



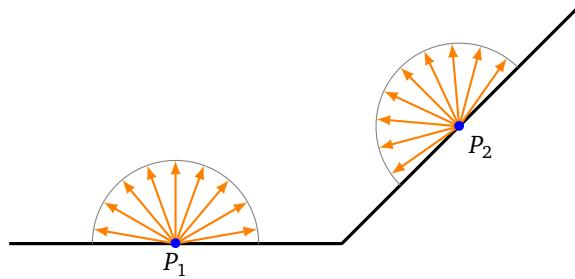
Lumière spéculaire



Lumière diffuse, ambiante et spéculaire

Luminosité ambiante.

La **luminosité ambiante** correspond à une lumière qui n'a pas de source précise et éclaire partout, et dans toutes les directions, de la même façon.



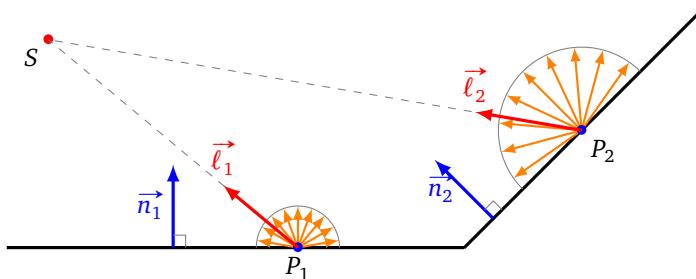
La couleur perçue Cl_{amb} dépend de la couleur Cl_{source} et de l'intensité i_{source} de l'éclairage, mais aussi de la couleur propre de l'objet Cl_{objet} :

$$Cl_{\text{amb}} = i_{\text{source}} Cl_{\text{source}} \otimes_{cl} Cl_{\text{objet}}$$

On rappelle que l'addition de deux couleurs, notée par \oplus_{cl} , se fait en ajoutant terme à terme les composantes r, g, b , sans dépasser 1. La multiplication, notée \otimes_{cl} , s'effectue en multipliant terme à terme les composantes r, g, b .

Luminosité diffuse.

La **luminosité diffuse** est une caractéristique associée aux matériaux mats et conduit à une réflexion de la lumière dans toutes les directions. Par contre, l'intensité des rayons réfléchis dépend de l'orientation de la surface par rapport au rayon lumineux.



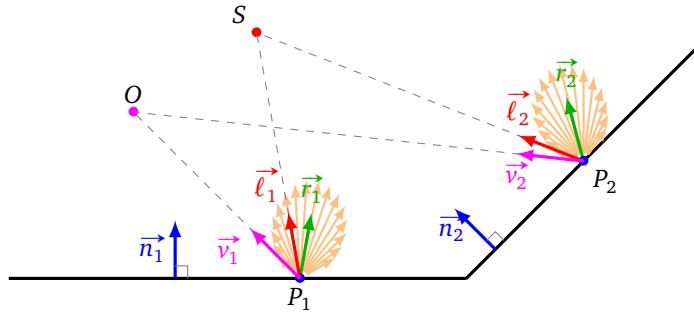
La formule de la couleur diffuse est résumée en :

$$Cl_{\text{diff}} = i_{\text{source}} \max(\vec{l} \cdot \vec{n}, 0) Cl_{\text{source}} \otimes_{cl} Cl_{\text{objet}}$$

où i_{source} et Cl_{source} sont l'intensité et la couleur de la source lumineuse et Cl_{objet} est la couleur propre de l'objet.

Luminosité spéculaire.

La **luminosité spéculaire** s'observe sur les objets brillants, les rayons lumineux se réfléchissent autour de l'axe principal de réflexion. La couleur perçue par luminosité spéculaire dépend de la position de l'observateur O .

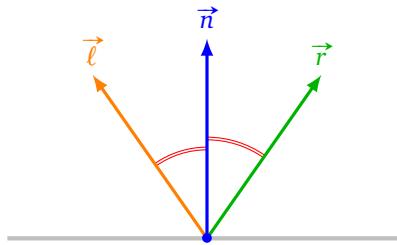


$$Cl_{\text{spec}} = i_{\text{source}} (\vec{r} \cdot \vec{v})^{e_{\text{spec}}} Cl_{\text{source}} \otimes_{cl} Cl_{\text{objet}}$$

La formule dépend :

- de l'intensité i_{source} et la couleur Cl_{source} de la source lumineuse mais aussi de la couleur Cl_{objet} propre à l'objet,
- de l'axe principal de réflexion \vec{r} ,
- de la direction \vec{v} vers l'observateur,
- d'un exposant d'étalement e_{spec} .

L'axe principal de la réflexion \vec{r} est le vecteur symétrique de \vec{l} (qui pointe vers la source lumineuse) par rapport à \vec{n} (orthogonal à la surface).



Éclairage direct. Cette couleur perçue, obtenue comme somme des trois composantes ambiante, diffuse, spéculaire, sera par la suite appelée la couleur obtenue par **éclairage direct**. Pour un point P d'un objet, observé depuis un point O , cette couleur sera notée :

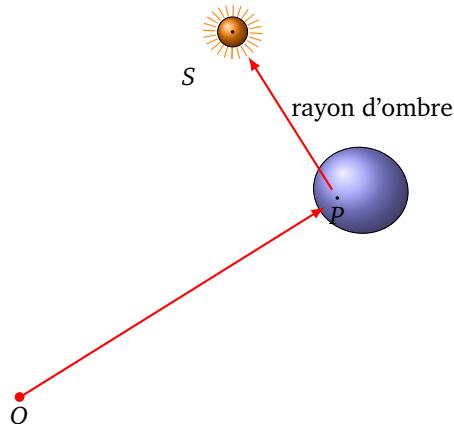
$$Cl_{P(\leftarrow O)}^{\text{dir}}$$

La flèche dans la notation « $P(\leftarrow O)$ » est dans le même sens que le rayon du *ray-tracing* qui va de l'observateur O vers le point P de l'objet (attention cette flèche va à rebours du rayon lumineux).

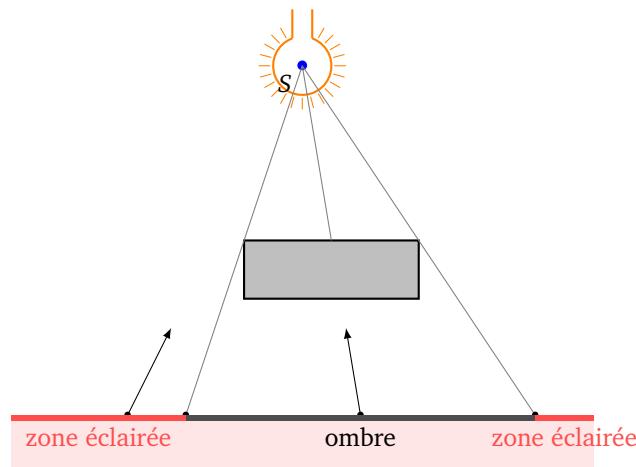
Nous allons maintenant apporter des améliorations à la version simplifiée du *ray-tracing* expliquée auparavant.

2.4. Ombre et sources lumineuses

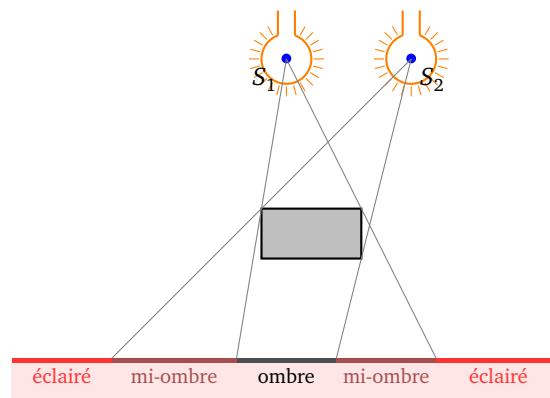
Le dernier rayon, lancé du point P de l'objet vers la source lumineuse S s'appelle le **rayon d'ombre** (*shadow ray*).



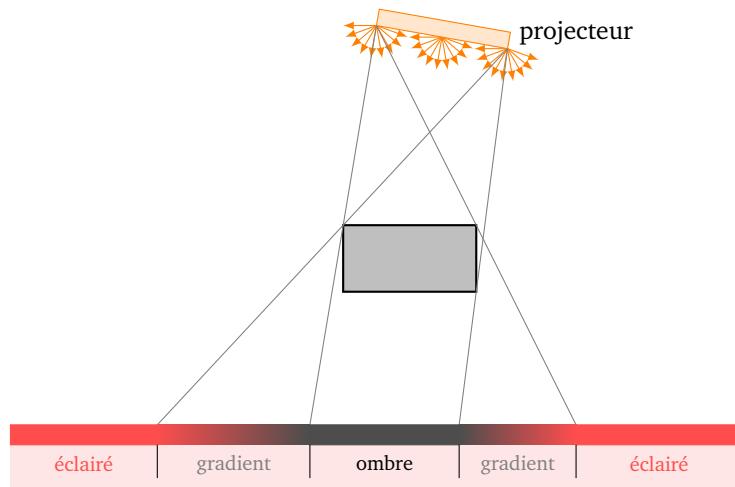
En effet, il se peut qu'un autre objet soit intercalé entre P et S . Si cet objet est opaque cela signifie que la source lumineuse est obstruée, autrement dit un rayon lumineux issu de S n'atteint pas le point P ; vous êtes dans l'ombre de l'objet et il n'y a pas d'éclairage direct ; en l'absence de toute autre source lumineuse, la couleur associée à ce point P est alors le noir. En général, on ajoute toujours une composante de lumière ambiante, ce qui fait que même un objet à l'ombre sera légèrement visible.



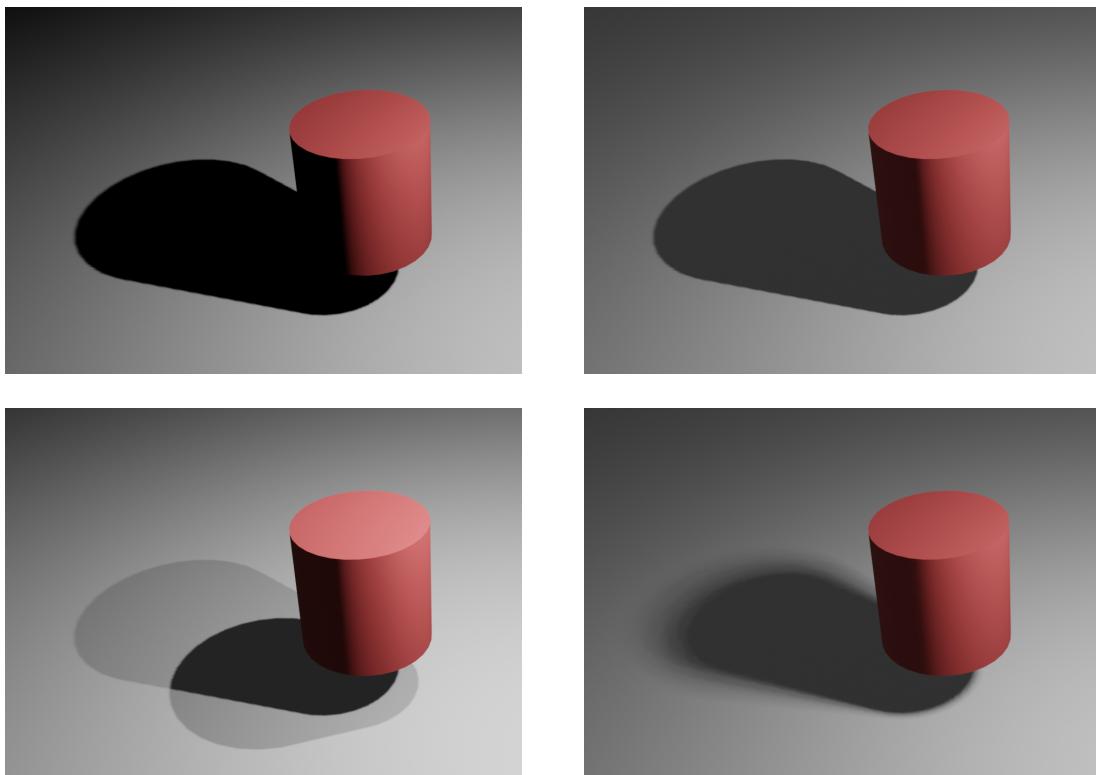
S'il y a plusieurs sources lumineuses, il faut lancer un rayon depuis P vers chacune des sources lumineuses $S_1, S_2\dots$. La couleur en P est alors la somme (au sens du chapitre « Lumière ») des couleurs résultant de chaque source.



Une source lumineuse peut ne pas être réduite à un point, cela peut être un disque (comme le Soleil apparent) ou un rectangle (comme un projecteur carré). Chaque point de cette surface peut être considéré comme une source de lumière ponctuelle ou directionnelle. L'intensité de la lumière qui atteint l'objet en un point P est proportionnelle à la surface visible de la source lumineuse depuis ce point.

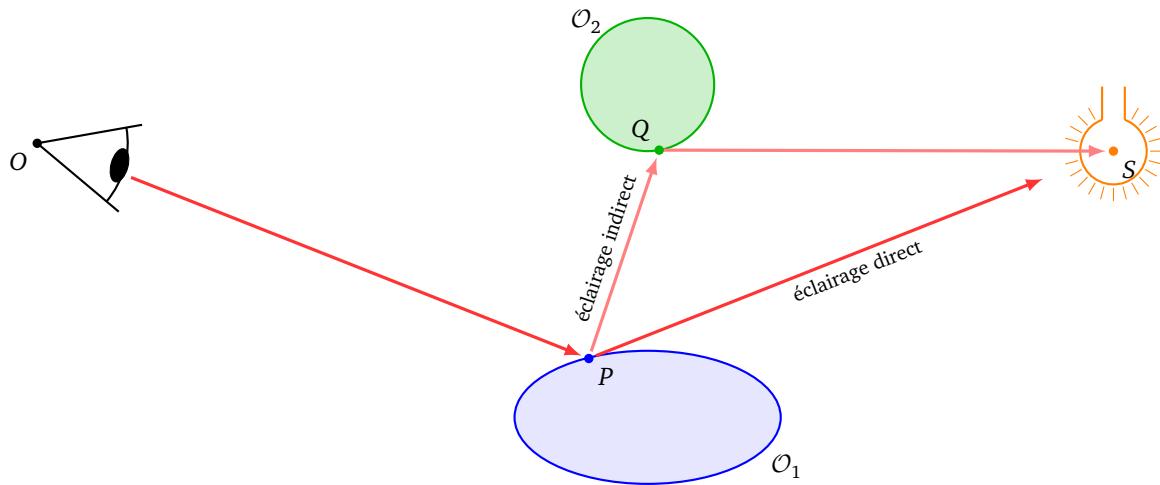


Ci-dessous, ligne du haut : (a) une source lumineuse sans lumière ambiante, (b) une source lumineuse avec une lumière ambiante. Ligne du bas : (c) deux sources lumineuses, (d) une source lumineuse non ponctuelle.



2.5. Réflexions multiples

Le *ray-tracing* prend toute sa force lorsqu'on considère les réflexions multiples qui donnent un éclairage indirect. Ainsi chaque objet peut être une source lumineuse pour les autres objets.



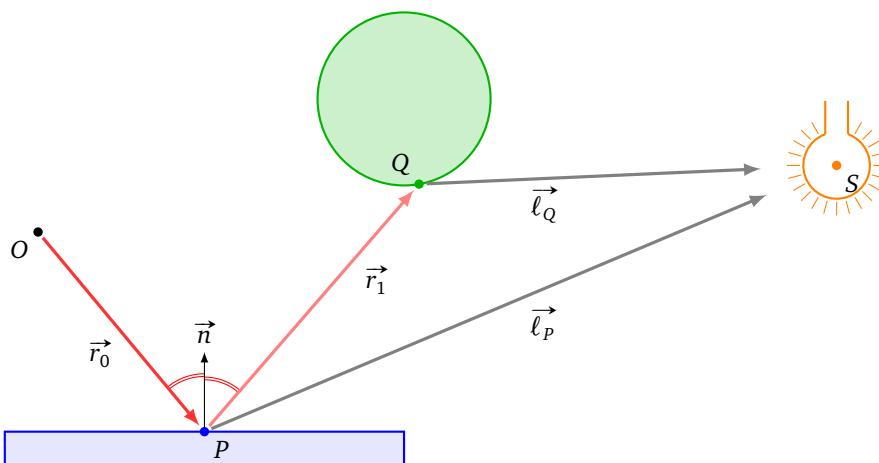
Dans la situation ci-dessus, le point \$P\$ de l'objet \$O_1\$ est éclairé de deux façons :

- par éclairage direct via un rayon issu de \$P\$ dirigé vers \$S\$ (comme auparavant),
- par éclairage indirect via l'objet \$O_2\$.

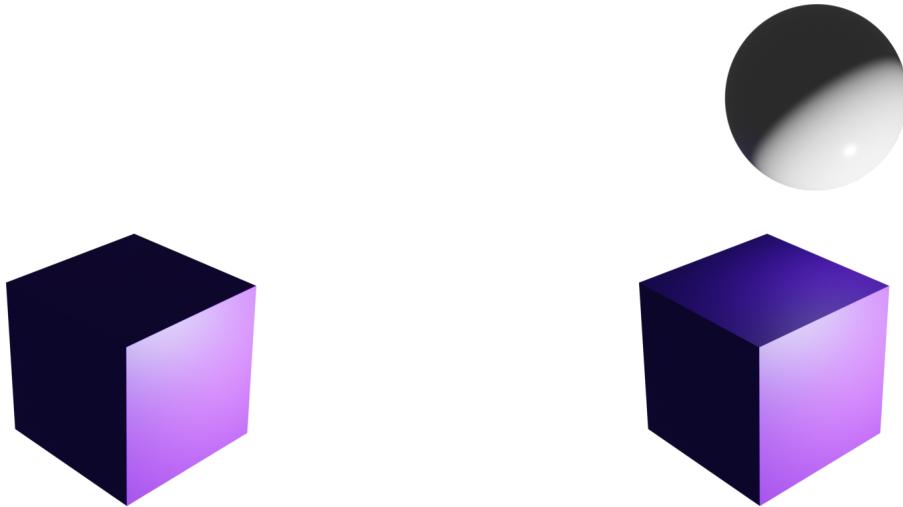
Expliquons comment calculer la composante issue de l'éclairage indirect.

1. On lance un rayon \$\vec{r}_0\$ issu de \$O\$, on calcule le premier point \$P\$ d'intersection avec un objet de la scène.
2. On calcule la luminosité associée à la lumière directe \$Cl_{P \leftarrow O}^{\text{dir}}\$ (en lançant un rayon de \$P\$ vers \$S\$).
3. On calcule le rayon \$\vec{r}_1\$ obtenu par une réflexion parfaite de \$\vec{r}_0\$ selon la normale en \$P\$. On calcule le premier point \$Q\$ d'intersection de \$\vec{r}_1\$ avec un objet de la scène.
4. On calcule la luminosité associée à la lumière directe en \$Q\$ perçue depuis \$P\$: \$Cl_{Q \leftarrow P}^{\text{dir}}\$ (en lançant un rayon de \$Q\$ vers \$S\$).
5. On calcule la composante de luminosité indirecte en \$P\$ issue de \$Q\$ comme une fraction de la composante précédente : \$Cl_{P \leftarrow Q}^{\text{indir}} = k_Q Cl_{Q \leftarrow P}^{\text{dir}}\$ où \$k_Q\$ est un coefficient, appelé *facteur de réflexivité*.
6. La luminosité totale perçue en \$P\$ est la somme de la luminosité directe et indirecte : \$Cl_P = Cl_P^{\text{dir}} \oplus_{cl} Cl_P^{\text{indir}}\$.

Souvenez-vous que les flèches « → » et « ← » indiquent le sens des rayons du *ray-tracing*.



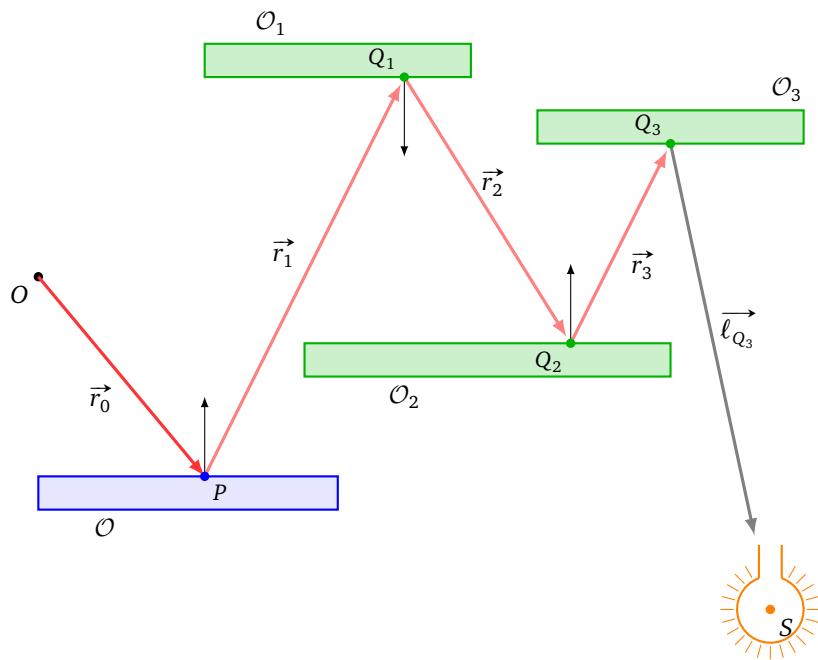
Ci-dessous à gauche, un cube avec un éclairage orienté vers le côté droit, la face du dessus et celle de gauche ne sont pas éclairées. Sur la figure de droite, on rajoute une sphère qui réfléchit la lumière, la face supérieure du cube est alors éclairée par une lumière indirecte.



Facteur de réflexivité. Le coefficient k_Q est un nombre réel entre 0 et 1 et dépend de la matière en Q . Un matériau absorbant aura un coefficient nul, un miroir aura un coefficient 1. Il est aussi important de garder à l'esprit qu'un rayon lumineux ne perd pas en intensité, quelle que soit la distance parcourue, tant qu'il ne rencontre pas d'obstacle. Notez bien que le rayon indirect (de P vers Q) est celui obtenu par réflexion parfaite. On pourrait considérer que ce modèle n'est pas réaliste et que la lumière se diffuse en Q , dans ce cas on diminue l'intensité perçue en Q proportionnellement au carré de la distance PQ .

Couleur au point de réflexion. Pour une meilleure modélisation il faut tenir compte de la couleur de la surface d'où provient la lumière indirecte. Ainsi une lumière blanche qui se réfléchit sur un objet vert en Q , va réfléchir une lumière verte vers P . Ceci est pris en compte dans le calcul de l'éclairage direct en Q , $Cl_{Q(\leftarrow P)}^{\text{dir}}$ pour lequel les formules des luminosités ambiante, diffuse et spéculaire tiennent compte de la couleur de l'objet Cl_{objet} en Q .

Réflexions multiples. Dans les explications précédentes nous n'avons considéré qu'un seul rebond indirect (au point Q). Pour les réflexions multiples il faut itérer le processus à partir du point Q . Sur le dessin ci-dessous la source lumineuse n'éclaire pas directement le point P , ni l'objet \mathcal{O}_1 (dans l'ombre), ni la face supérieure de l'objet \mathcal{O}_2 . Seul l'objet \mathcal{O}_3 est éclairé directement.



Pour calculer l'éclairage indirect en \$P\$, vu de \$O\$:

- on lance un rayon \$\vec{r}_0\$ de \$O\$ vers \$P\$,
- par symétrie par rapport à la normale en \$P\$, on lance un rayon \$\vec{r}_1\$ de \$P\$ vers un point \$Q_1\$,
- par symétrie par rapport à la normale en \$Q_1\$, on lance un rayon \$\vec{r}_2\$ de \$Q_1\$ vers un point \$Q_2\$,
- par symétrie par rapport à la normale en \$Q_2\$, on lance un rayon \$\vec{r}_3\$ de \$Q_2\$ vers un point \$Q_3\$,
- depuis \$Q_3\$ on lance un rayon d'ombre \$\overrightarrow{\ell}_{Q_3}\$ vers la source lumineuse \$S\$.

On remonte ensuite notre construction : on calcule la luminosité directe perçue en \$Q_3\$ (depuis \$Q_2\$) : \$Cl_{Q_3}\$.

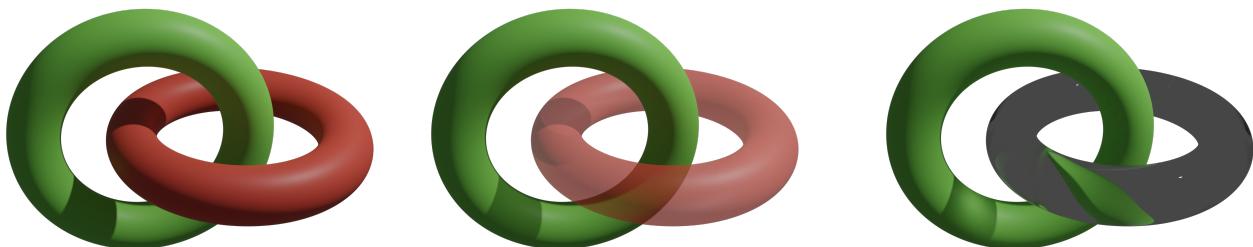
Ainsi la luminosité indirecte perçue en \$Q_2\$ est \$k_{Q_3} Cl_{Q_3}\$, celle en \$Q_1\$ est \$k_{Q_2} k_{Q_3} Cl_{Q_3}\$ et au final la luminosité indirecte perçue en \$P\$ est :

$$Cl_P^{\text{indir}} = k_{Q_1} k_{Q_2} k_{Q_3} Cl_{Q_3}.$$

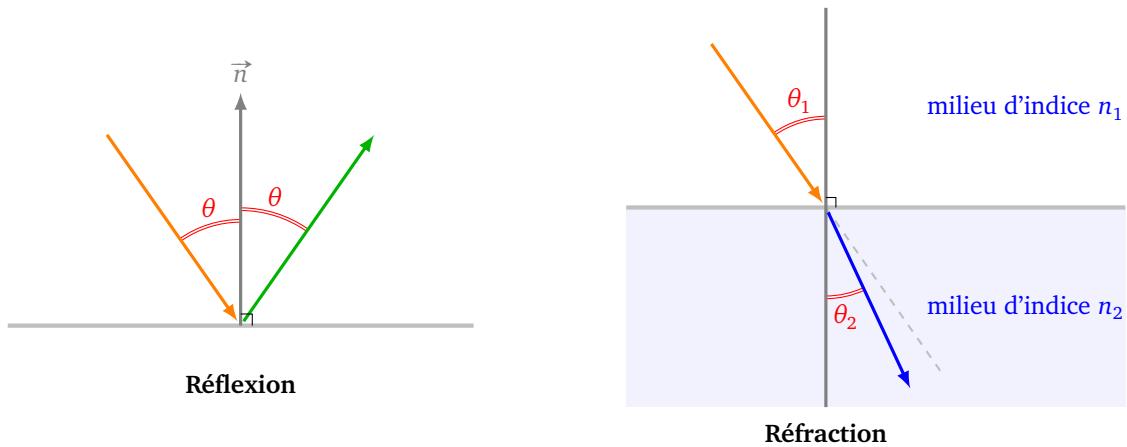
Dans cet exemple on a considéré que les objets intercalés ne changent pas la couleur de la lumière mais juste son intensité. Nous étudierons la formule de récurrence générale juste après. Comme chaque coefficient \$k\$ est inférieur à 1, l'intensité de la lumière indirecte diminue à chaque réflexion et devient vite négligeable après 1, 2 ou 3 réflexions. Il existe cependant une exception : dans le cas d'un miroir parfait où \$k = 1\$, on peut avoir (en théorie) une infinité de réflexions ; c'est un phénomène que vous avez sûrement déjà expérimenté en vous plaçant entre deux miroirs parallèles.

2.6. Transparency

Nous allons étudier la propriété de transparence. Ci dessous : (a) deux tores opaques, (b) un seul tore transparent (sans changement milieu), (c) un tore est en verre (d'indice \$n \approx 1.4\$).



Nous avons déjà étudié la **réflexion** d'un rayon dans le chapitre « Lumière » : un rayon arrivant sur une surface rebondit dans la direction symétrique par rapport à la normale.

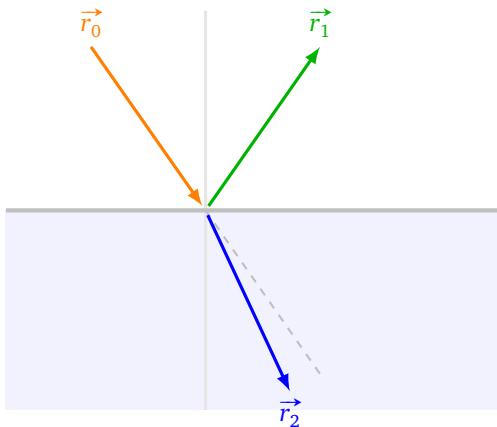


Nous étudierons plus tard la **réfraction** (voir le chapitre « Physique ») : lorsqu'un rayon change de milieu, par exemple il passe de l'air à l'eau ou bien traverse du verre, sa trajectoire est déviée. L'angle de déviation est régi par la loi de Snell-Descartes :

$$n_1 \sin \theta_1 = n_2 \sin \theta_2$$

où n_1 et n_2 sont les *indices* qui dépendent du milieu (mais aussi de la longueur d'onde du rayon).

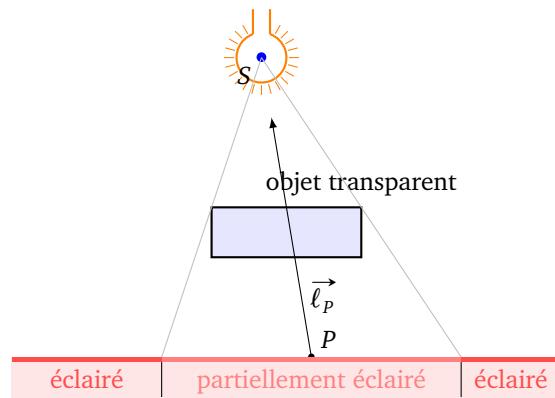
Pour un matériau transparent, par exemple du verre, les deux phénomènes se produisent : une partie du rayon est réfléchie, l'autre est réfractée. De plus une partie de l'énergie du rayon est absorbée par le matériau (ce qui se traduit dans nos calculs par le facteur $k \leq 1$).



Rayon réfléchi et rayon réfracté

Conclusion. Dorénavant lorsqu'un rayon atteint un objet ayant des propriétés de transparence il faut relancer deux rayons depuis le point d'impact : un rayon réfléchi et un rayon réfracté.

Il y a une exception que l'on fera : lorsqu'on lance le rayon d'ombre \vec{l}_P , on considère qu'il se dirige tout droit vers la source lumineuse, si ce rayon rencontre un objet translucide on peut éventuellement atténuer la luminosité.



Exercice.

1. Sur la figure de gauche ci-dessous, tracer les rayons correspondant à l'éclairage direct d'un point de la surface du lac, puis faire la même chose avec un point de la surface de la montagne.

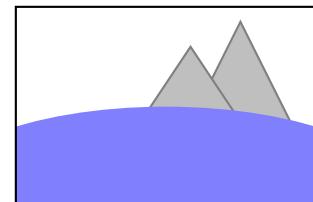
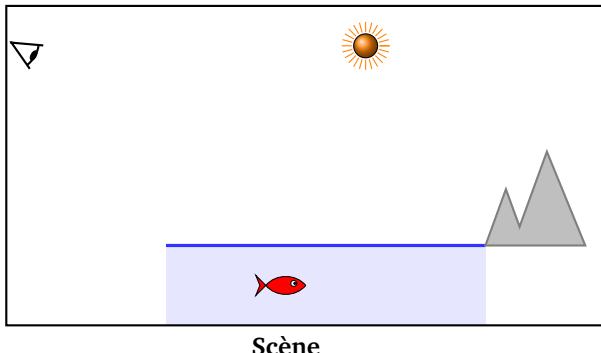


Image – Éclairage direct

2. Sur la figure de gauche ci-dessous, tracer les rayons correspondant à l'éclairage indirect d'un point de la surface du lac où apparaît le reflet de la montagne.

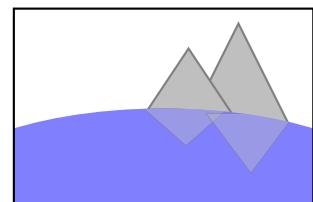
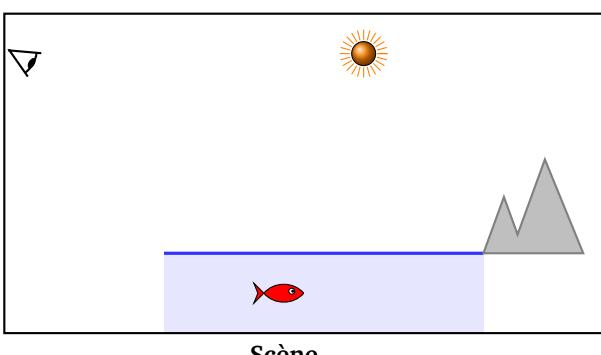


Image – Éclairage indirect

3. Sur la figure de gauche ci-dessous, tracer les rayons correspondant à l'éclairage du poisson par transparence.

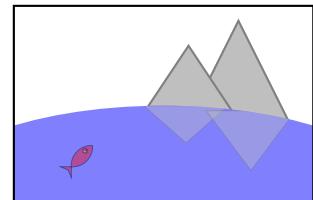
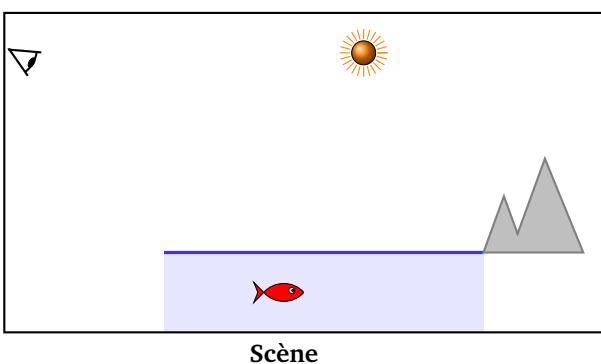
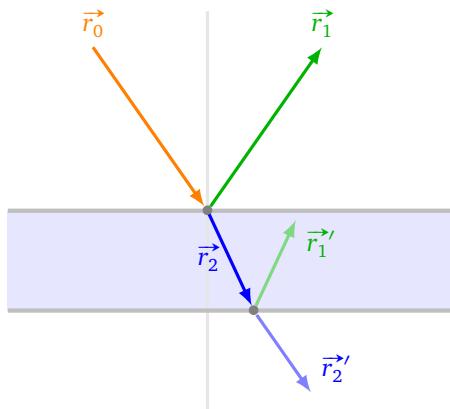


Image – Transparence

Complications.

Paroi. Lorsque le rayon traverse une paroi de verre alors il faut appliquer ce principe des deux côtés de la paroi. Noter que le rayon traversant la paroi ressort parallèlement au rayon incident.



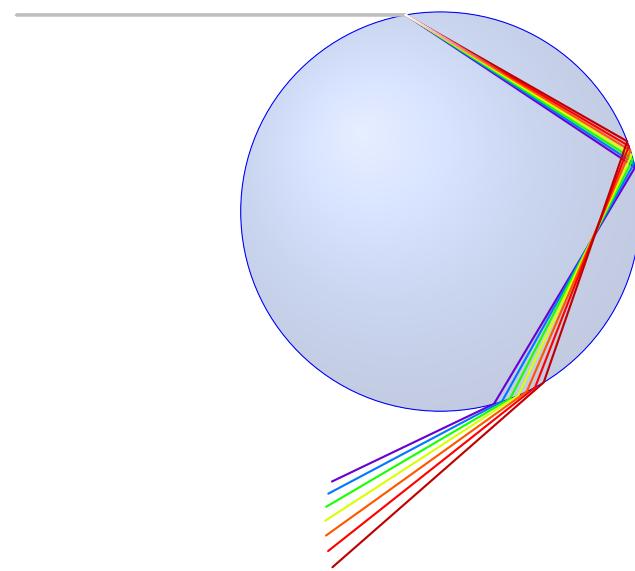
Rayon traversant une paroi de verre

Réflexion totale. Lorsqu'on lance un rayon sous l'eau avec un angle faible par rapport à l'horizontale le comportement est différent : le rayon est complètement réfléchi, c'est le phénomène de **réflexion totale** que vous pouvez observer du fond de l'eau en levant la tête vers la surface qui a alors l'allure d'un miroir.



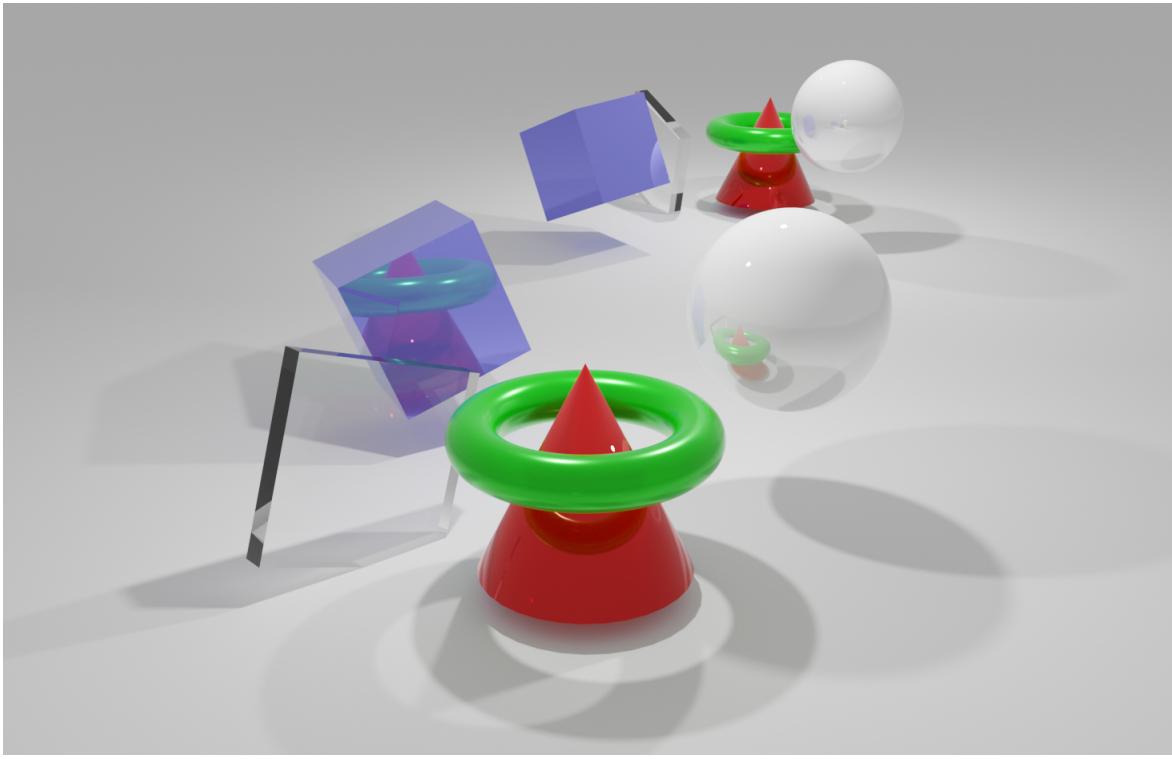
Réflexion totale

Ces phénomènes de réflexion/réfraction expliquent la formation des arcs-en-ciel, en effet les rayons se réfractent différemment selon leur couleur. La lumière blanche du Soleil, constituée de rayons de toutes les couleurs, entre dans une goutte d'eau, les rayons sont plus ou moins réfractés selon leur longueur d'onde, il y a ensuite une réflexion interne à la goutte puis une nouvelle réfraction qui sépare davantage les rayons. On retrouvera ce phénomène dans le chapitre « Physique » avec l'expérience du prisme de Newton.



2.7. Mise en œuvre

Expliquons les grandes lignes de l'algorithme de *ray-tracing*. C'est un excellent projet de le programmer mais cela demande plusieurs jours de travail. Ci-dessous une scène avec cinq objets, dont un transparent, devant un miroir, avec deux sources lumineuses.



Au préalable nous devons disposer de :

- une fonction `lancer_rayon(O, r)` qui calcule les points d'intersection d'un rayon \vec{r}_0 issu de O avec les objets de la scène et renvoie le point P le plus proche,
- une fonction `couleur_directe(P, 0)` qui renvoie la couleur $Cl_{P(\leftarrow O)}^{\text{dir}}$ en P perçue de O , obtenue par addition des couleurs ambiante, diffuse et spéculaire.

On souhaite attribuer une couleur au pixel E_{ij} de l'écran, pour cela on calcule le rayon \vec{r}_0 de O vers E_{ij} . On note P le premier point d'intersection de \vec{r}_0 avec la scène, si le rayon ne coupe aucun objet on colorie le pixel par la couleur de fond.

Il s'agit ensuite de définir une fonction `couleur(P, 0)` qui calcule la couleur $Cl_{P(\leftarrow O)}$ obtenue par *ray-tracing*. Cette fonction est une fonction récursive (elle s'appelle elle-même).

Structure de `couleur(P, 0)` :

1. On teste la condition d'arrêt.
2. On calcule la couleur directe $Cl_{P(\leftarrow O)}^{\text{dir}}$ en P perçue de O .
3. on note \vec{r}_0 le rayon de O vers P et \vec{r}_1 le rayon obtenu par réflexion parfaite en P ; on calcule Q_1 le premier point d'intersection de \vec{r}_1 avec la scène.
4. Par un appel récursif `couleur(Q1, P)`, on calcule la couleur $Cl_{Q_1(\leftarrow P)}$ en Q_1 perçue depuis P .
5. On calcule la couleur indirecte en P issue de Q_1 :

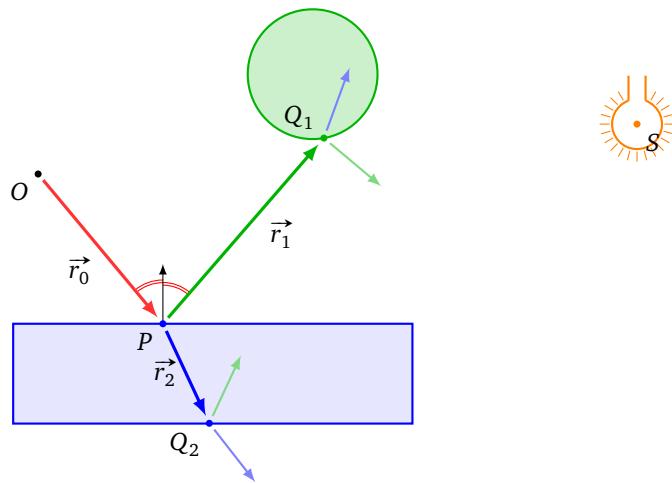
$$Cl_{P(\rightarrow Q_1)}^{\text{indir}} = k_{Q_1} Cl_{Q_1(\leftarrow P)}.$$

6. Si l'objet a des propriétés de transparence on note \vec{r}_2 le rayon obtenu par réfraction de \vec{r}_0 en P , on calcule Q_2 le premier point d'intersection de \vec{r}_2 avec la scène.
7. Par un appel récursif `couleur(Q2, P)` on calcule la couleur $Cl_{Q_2(\leftarrow P)}$ en Q_2 perçue depuis P .
8. On calcule la couleur indirecte en P issue de Q_2 :

$$Cl_{P(\rightarrow Q_2)}^{\text{indir}} = k_{Q_2} Cl_{Q_2(\leftarrow P)}.$$

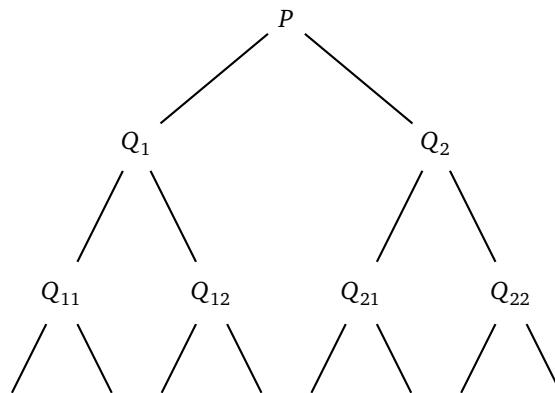
9. On renvoie la couleur $Cl_{P(\leftarrow O)}$ comme somme de la couleur directe et des couleurs indirectes :

$$Cl_{P(\leftarrow O)} = Cl_{P(\leftarrow O)}^{\text{dir}} \oplus_{cl} Cl_{P(\rightarrow Q_1)}^{\text{indir}} \oplus_{cl} Cl_{P(\rightarrow Q_2)}^{\text{indir}}.$$



Condition(s) d'arrêt. Les rayons pouvant rebondir une infinité de fois, la condition d'arrêt est importante. La condition la plus simple est de définir à l'avance un nombre maximal de rebonds possibles, autrement dit de limiter la profondeur des appels récursifs. On peut en plus stopper le processus dès que la luminosité indirecte va être faible et ne contribue presque plus à l'éclairage. On peut par exemple arrêter les appels récursifs dès que le produit des constantes de réflexivité est suffisamment petit : $k_{Q_{i_1}} k_{Q_{i_2}} \cdots k_{Q_{i_n}} \leq \epsilon$.

Arbre de récursivité. L'arbre suivant montre la succession des points où il faut calculer la lumière directe afin d'obtenir l'éclairage indirect en P .

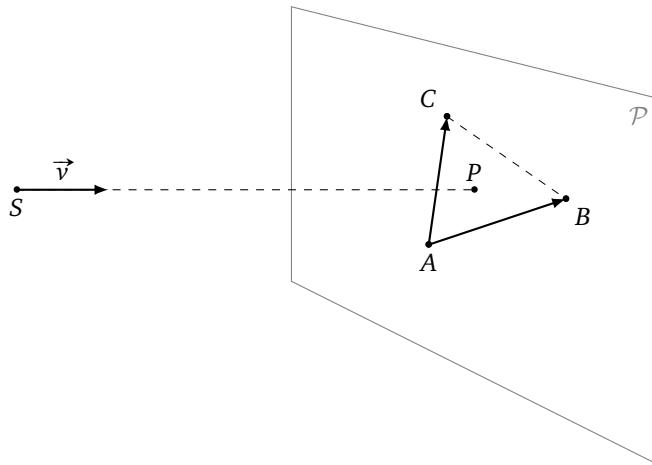


3. Intersection efficace avec un triangle

Dans le chapitre « Lancer de rayons I » nous avons vu comment calculer l'intersection d'un rayon avec des objets géométriques simples. Les figures géométriques les plus importantes sont les triangles de l'espace qui correspondent aux faces d'un maillage triangulaire. Nous allons donc calculer plus efficacement l'intersection d'un rayon avec un triangle.

3.1. Rayon lancé sur un triangle

Considérons un triangle de l'espace défini par trois points A, B, C non alignés. On lance un rayon issu de S dans la direction \vec{v} . On suppose que le rayon n'est pas parallèle au plan \mathcal{P} contenant le triangle. Notons P le point d'intersection du rayon avec ce plan \mathcal{P} .



Il existe donc $t \in \mathbb{R}$ tel que :

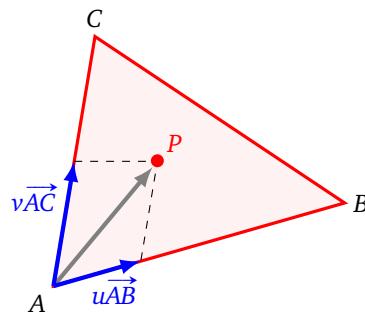
$$P = S + t \vec{v}$$

et comme P est dans le plan (ABC) , il existe $u, v \in \mathbb{R}$ tels que :

$$P = A + u \vec{AB} + v \vec{AC}.$$

Le point P est situé à l'intérieur (ou sur le bord) du triangle ABC si et seulement si $0 \leq u \leq 1$ et $0 \leq v \leq 1$.

Pour les explications sur les coordonnées barycentriques $(1-u-v : u : v)$ on renvoie au chapitre « Texture ».



Nous trouvons t, u, v en résolvant le système linéaire de 3 équations et 3 inconnues, donné par :

$$S + t \vec{v} = A + u \vec{AB} + v \vec{AC}$$

En écrivant les coordonnées :

$$\begin{pmatrix} x_S \\ y_S \\ z_S \end{pmatrix} + t \begin{pmatrix} x_v \\ y_v \\ z_v \end{pmatrix} = \begin{pmatrix} x_A \\ y_A \\ z_A \end{pmatrix} + u \begin{pmatrix} x_B - x_A \\ y_B - y_A \\ z_B - z_A \end{pmatrix} + v \begin{pmatrix} x_C - x_A \\ y_C - y_A \\ z_C - z_A \end{pmatrix}.$$

Notons

$$M = \begin{pmatrix} x_v & -(x_B - x_A) & -(x_C - x_A) \\ y_v & -(y_B - y_A) & -(y_C - y_A) \\ z_v & -(z_B - z_A) & -(z_C - z_A) \end{pmatrix} \quad X = \begin{pmatrix} t \\ u \\ v \end{pmatrix} \quad Y = \begin{pmatrix} x_A - x_S \\ y_A - y_S \\ z_A - z_S \end{pmatrix}$$

Le système linéaire équivaut à l'équation matricielle, d'inconnue X :

$$MX = Y.$$

On peut alors mettre en œuvre différentes techniques d'algèbre linéaire :

1. calcul de l'inverse M^{-1} , puis $X = M^{-1}Y$,
2. la méthode du pivot de Gauss,
3. les formules de Cramer.

C'est cette dernière méthode que nous allons détailler, mais auparavant rappelons comment calculer un déterminant.

3.2. Déterminant 3×3

Rappelons la formule du déterminant, uniquement dans le cas d'une matrice 3×3 .

Si M est une matrice 3×3 :

$$M = \begin{pmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ a_{31} & a_{32} & a_{33} \end{pmatrix}$$

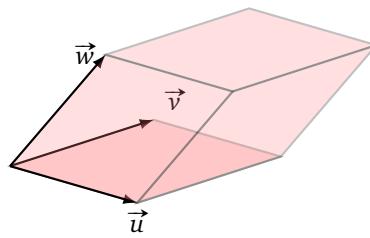
alors le déterminant se calcule selon la formule :

$$\det(M) = a_{11}a_{22}a_{33} + a_{12}a_{23}a_{31} + a_{13}a_{21}a_{32} - a_{31}a_{22}a_{13} - a_{32}a_{23}a_{11} - a_{33}a_{21}a_{12}.$$

Il existe d'autres façons de calculer un déterminant, on renvoie à un cours d'algèbre linéaire.

Rappelons cependant l'interprétation géométrique en termes de vecteurs. Soient trois vecteurs de l'espace $\vec{u}, \vec{v}, \vec{w}$. On forme la matrice M de taille 3×3 en juxtaposant les vecteurs $\vec{u}, \vec{v}, \vec{w}$ considérés comme des vecteurs colonnes.

$$M = \begin{pmatrix} \vec{u} & \vec{v} & \vec{w} \\ a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ a_{31} & a_{32} & a_{33} \end{pmatrix}$$



Le déterminant de M , que l'on peut aussi noter $\det(\vec{u}, \vec{v}, \vec{w})$, est égal au volume du parallélépipède formé par les trois vecteurs.

La formule du **produit mixte** (*triple product*) redonne aussi le déterminant :

$$\det(\vec{u}, \vec{v}, \vec{w}) = \vec{u} \cdot (\vec{v} \wedge \vec{w}).$$

Il faut donc d'abord calculer un produit vectoriel puis un produit scalaire. L'ordre des vecteurs est important pour le signe du déterminant.

3.3. Méthode de Cramer

La **règle de Cramer** est une formule qui donne la solution d'un système linéaire ayant autant d'équations que d'inconnues. On se contente de l'expliquer dans le cas 3×3 . Considérons le système d'équations linéaires à 3 équations et 3 inconnues suivant :

$$\begin{cases} a_{11}x_1 + a_{12}x_2 + a_{13}x_3 = y_1 \\ a_{21}x_1 + a_{22}x_2 + a_{23}x_3 = y_2 \\ a_{31}x_1 + a_{32}x_2 + a_{33}x_3 = y_3 \end{cases}$$

Ce système peut aussi s'écrire sous forme matricielle $MX = Y$ où

$$M = \begin{pmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ a_{31} & a_{32} & a_{33} \end{pmatrix} \in M_3(\mathbb{R}), \quad X = \begin{pmatrix} x_1 \\ x_2 \\ x_3 \end{pmatrix} \quad \text{et} \quad Y = \begin{pmatrix} y_1 \\ y_2 \\ y_3 \end{pmatrix}.$$

Définissons les matrices M_1, M_2, M_3 en remplaçant la j -ème colonne de M par le second membre Y :

$$M_1 = \begin{pmatrix} Y & a_{12} & a_{13} \\ y_1 & a_{22} & a_{23} \\ y_2 & a_{32} & a_{33} \\ y_3 & & \end{pmatrix} \quad M_2 = \begin{pmatrix} a_{11} & Y & a_{13} \\ a_{21} & y_1 & a_{23} \\ a_{31} & y_2 & a_{33} \\ y_3 & & \end{pmatrix} \quad M_3 = \begin{pmatrix} a_{11} & a_{12} & Y \\ a_{21} & a_{22} & y_1 \\ a_{31} & a_{32} & y_2 \\ y_3 & & \end{pmatrix}$$

La règle de Cramer va nous permettre de calculer la solution du système dans le cas où $\det M \neq 0$ en fonction des déterminants des matrices M et M_j .

Théorème 1 (Règle de Cramer).

Soit $MX = Y$ un système de 3 équations à 3 inconnues. Supposons que $\det M \neq 0$. Alors l'unique solution (x_1, x_2, x_3) du système est donnée par :

$$x_1 = \frac{\det M_1}{\det M} \quad x_2 = \frac{\det M_2}{\det M} \quad x_3 = \frac{\det M_3}{\det M}.$$

Exemple.

Résolvons le système suivant :

$$\begin{cases} -2x_1 + 4x_2 - x_3 = 10 \\ x_1 + 3x_3 = 3 \\ x_1 - 2x_2 + 3x_3 = 4. \end{cases}$$

On a

$$M = \begin{pmatrix} -2 & 4 & -1 \\ 1 & 0 & 3 \\ 1 & -2 & 3 \end{pmatrix} \quad Y = \begin{pmatrix} 10 \\ 3 \\ 4 \end{pmatrix}$$

$$M_1 = \begin{pmatrix} 10 & 4 & -1 \\ 3 & 0 & 3 \\ 4 & -2 & 3 \end{pmatrix} \quad M_2 = \begin{pmatrix} -2 & 10 & -1 \\ 1 & 3 & 3 \\ 1 & 4 & 3 \end{pmatrix} \quad M_3 = \begin{pmatrix} -2 & 4 & 10 \\ 1 & 0 & 3 \\ 1 & -2 & 4 \end{pmatrix}$$

et

$$\det M = -10 \quad \det M_1 = 78 \quad \det M_2 = 5 \quad \det M_3 = -36.$$

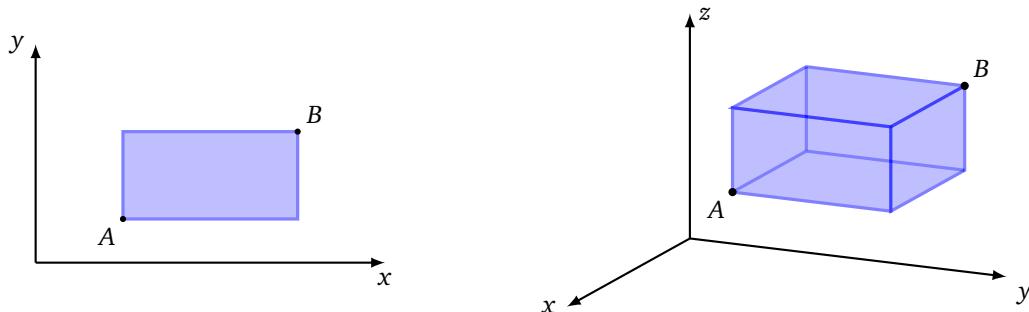
La solution est alors :

$$x_1 = \frac{\det M_1}{\det M} = \frac{78}{-10} = -\frac{39}{5} \quad x_2 = \frac{\det M_2}{\det M} = \frac{5}{-10} = -\frac{1}{2} \quad x_3 = \frac{\det M_3}{\det M} = \frac{-36}{-10} = \frac{18}{5}.$$

4. Boites englobantes

4.1. Principe

Calculer l'intersection d'un rayon avec un objet géométrique, ou même juste savoir si cette intersection existe, peut être coûteux en calculs. On renvoie par exemple au calcul de l'intersection d'un rayon avec une sphère, expliqué dans le chapitre « Lancer de rayons I ». Heureusement pour certains objets il est très facile de décider si le rayon coupe ou pas cet objet : le meilleur exemple est le cas d'une boite. Rappelons qu'une boite 2D est simplement un rectangle dont les bords sont parallèles aux axes horizontaux et verticaux. Une boite 3D est un parallélépipède dont les faces sont parallèles aux plans de coordonnées. Nous avons vu, toujours dans le chapitre « Lancer de rayons I », comment calculer l'intersection d'un rayon et d'une boite.

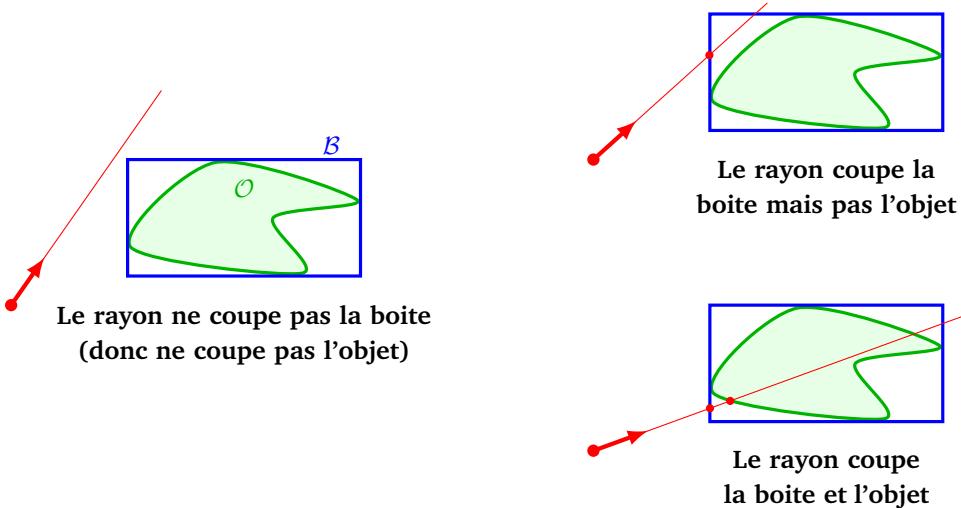


Considérons un objet \mathcal{O} d'une scène. On lance des rayons. Si l'objet n'est pas trop gros, la plupart des rayons ne vont pas couper l'objet. On souhaite donc écarter rapidement un maximum de rayons inutiles. L'idée est

la suivante : on englobe l'objet \mathcal{O} dans une boîte \mathcal{B} . Il est évident que :

Si le rayon ne coupe pas la boîte \mathcal{B} , alors il ne coupe pas l'objet \mathcal{O} .

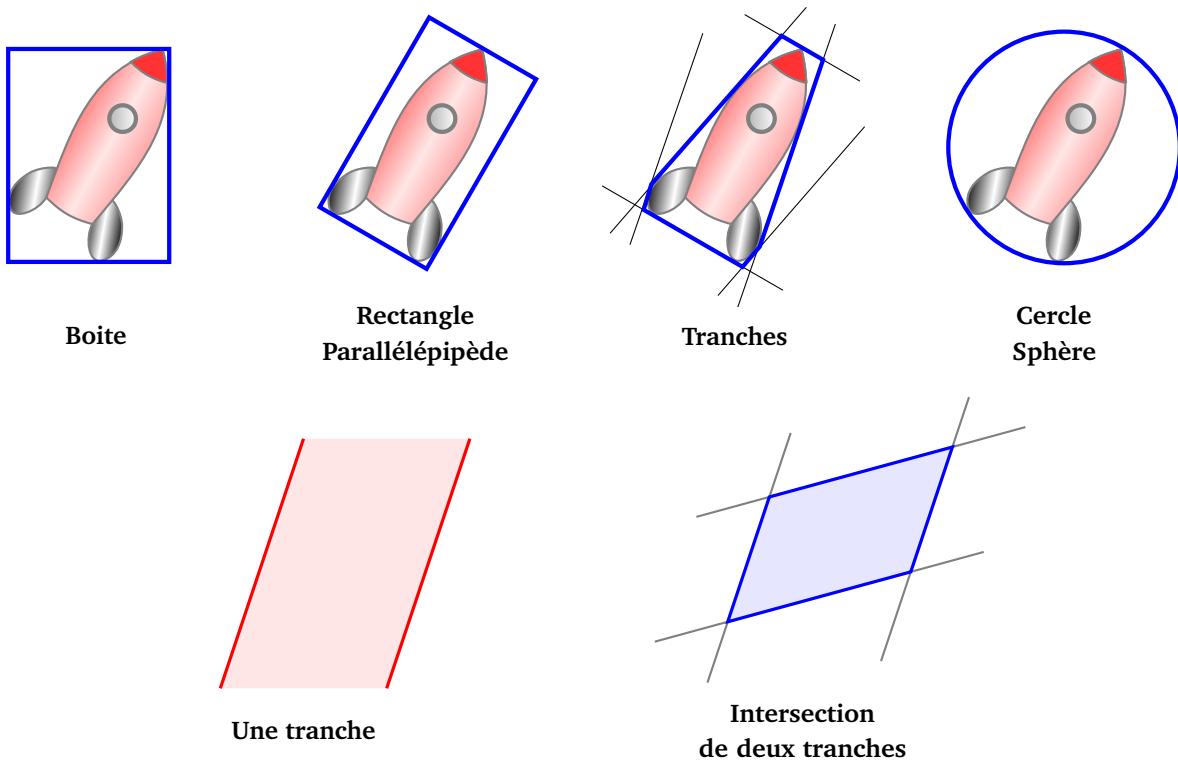
Bien sûr la réciproque n'est pas vraie. Donc si un rayon coupe la boîte, il faut se lancer dans les calculs plus compliqués pour savoir si le rayon coupe l'objet ou pas. Évidemment on a tout intérêt à ce que la boîte englobante soit la plus adaptée possible en minimisant ses dimensions pour « coller » à l'objet.



4.2. Différents volumes englobants

Voici différentes types de volumes englobants :

- une boîte,
- un rectangle (2D) ou un parallélépipède rectangle (3D) (dont les faces ne sont pas nécessairement parallèles aux axes),
- une intersection de tranches. Une *tranche* (*slab*) est la région du plan comprise entre deux droites parallèles (dans l'espace c'est la région comprise entre deux plans parallèles),
- un cercle ou une sphère.



On pourrait imaginer des polygones ou polyèdres convexes (on renvoie au chapitre « Triangulation » pour le calcul de l'enveloppe convexe).

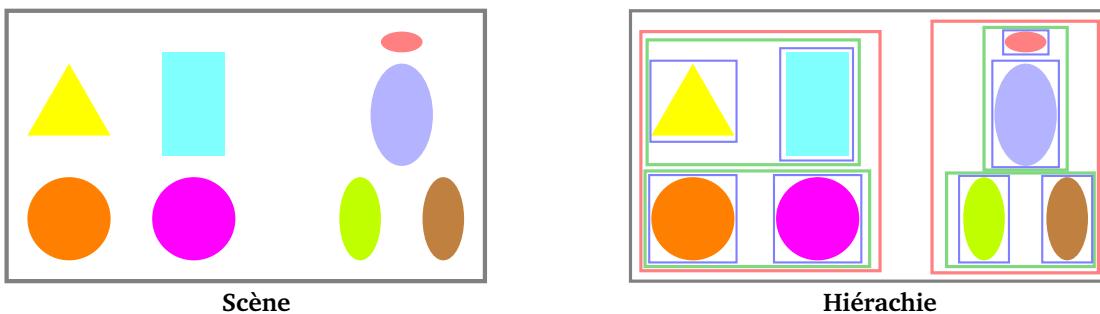
Le plus utilisé est aussi le plus simple : la boîte. L'encodage d'une boîte par la convention « AABB » correspond aux coordonnées du coin inférieur gauche A et du coin supérieur droit B du rectangle, il y a simplement 4 réels (x_A, y_A, x_B, y_B). En dimension 3, on prend les coordonnées de deux sommets sur une diagonale, il y a 6 réels ($x_A, y_A, z_A, x_B, y_B, z_B$). (Voir les dessins ci-dessus du paragraphe « Principe ».)

4.3. Hiérarchie

L'idée de volume englobant prend davantage d'intérêt pour les scènes complexes où l'on peut alors englober les boîtes dans d'autres boîtes afin de retarder au maximum les calculs compliqués. Si la scène est composée de beaucoup d'objets, un rayon va couper seulement une petite partie d'entre eux, et donc on cherche à écarter rapidement un maximum d'objets qui ne coupent pas ce rayon.

Nous allons construire une **hiérarchie** (BVH, *Bounding Volume Hierarchy*) :

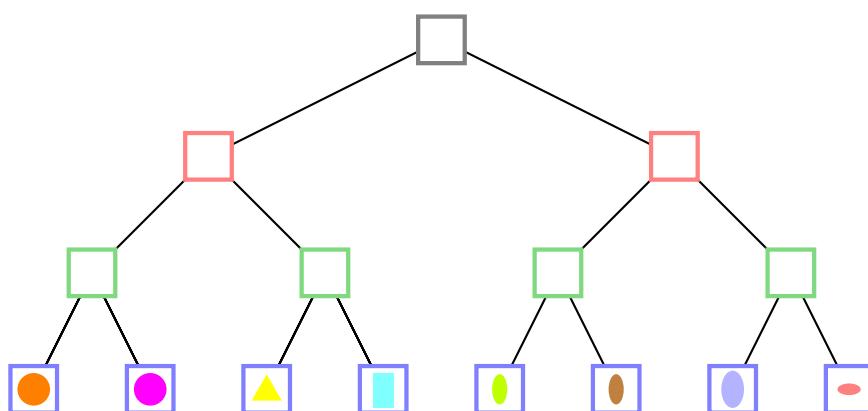
- On sépare les objets de la scène en deux parties, par exemple on englobe les objets de gauche dans un boîte et les objets de droite dans une autre boîte (quitte à découper en deux un objet qui se trouverait au milieu).
- On sépare les objets de gauche dans deux boîtes une boîte en haut à gauche, une boîte en bas à gauche. On fait de même avec la boîte de droite.
- On itère jusqu'à avoir un seul objet par boîte.



On représente ces inclusions de boîtes par un arbre binaire (chaque sommet a au plus deux enfants) : une boîte correspond à un sommet et les boîtes incluses sont les descendants de ce sommet. Les feuilles sont les boîtes ne contenant qu'un seul objet.

Lorsqu'on lance un rayon, on teste s'il coupe une boîte :

- si ce n'est pas le cas, le rayon ne coupe aucun objet inclus dans la boîte,
- si c'est le cas on teste l'intersection du rayon avec les deux sous-boîtes...



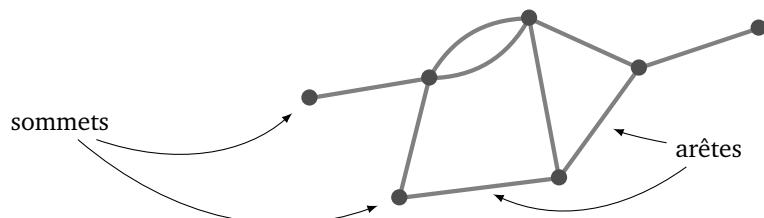
Triangulation

Nous découpons le plan en objets simples : des triangles.

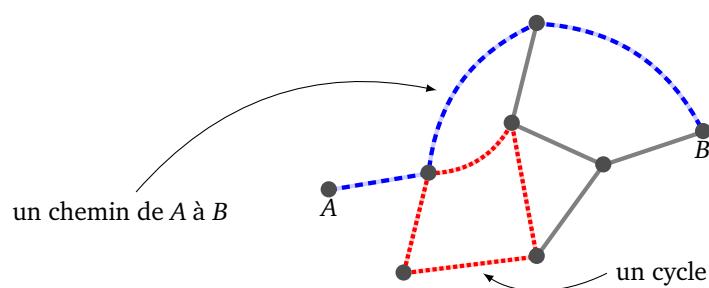
1. Graphes

1.1. Vocabulaire

- Un **graphe** est un ensemble de **sommets** et d'**arêtes**. Une arête commence à un sommet et se termine à un sommet, on dit alors que les sommets sont **adjacents**.



- Un **chemin** est une suite d'arêtes consécutives (ces arêtes étant toutes distinctes). Un **cycle** est un chemin qui commence et termine au même sommet.

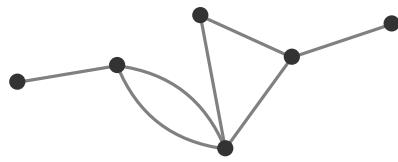


- Un graphe est **orienté** lorsqu'on définit un sens à chaque arête.

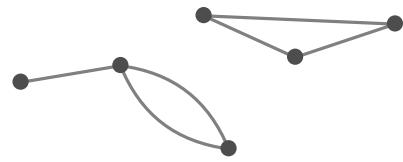


Pour le graphe orienté ci-dessus il existe un chemin qui va de A à B , mais pas de chemin qui va de B à A .

- Un graphe est **connexe** si entre deux sommets quelconques, il existe un chemin les reliant.

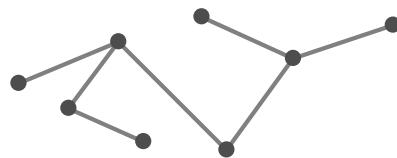


Graphe connexe

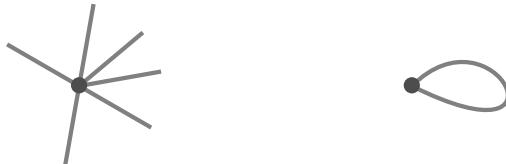


Graphe non connexe

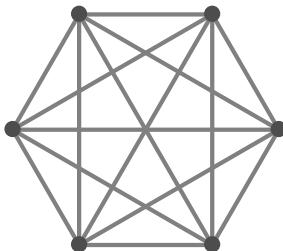
- Un **arbre** est un graphe connexe sans cycle. Dans un arbre, entre deux sommets quelconques, il existe un unique chemin reliant ces deux sommets.



- Le **degré** d'un sommet est le nombre d'arêtes issues de ce sommet. Une **boucle** est une arête reliant un sommet à lui-même (elle compte double pour le degré). Ci-dessous un sommet de degré 6 (à gauche) et une boucle (à droite) :



- Un graphe est **complet**, si entre deux sommets distincts A et B quelconques il existe une unique arête reliant A à B .

Le graphe complet K_6

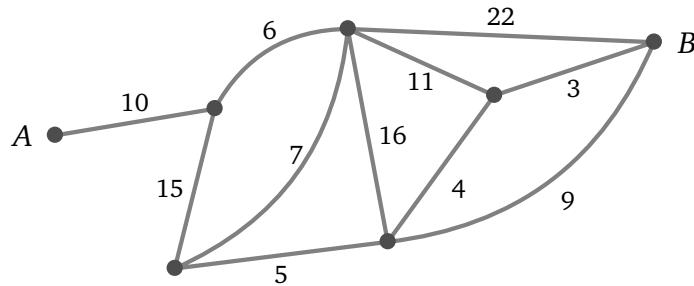
1.2. Problèmes en lien avec des graphes

Problème du voyageur de commerce.

Un voyageur doit visiter une liste de villes, il souhaite relier toutes ces villes en effectuant le trajet le plus court possible. Formalisons ce problème en termes de graphe : une ville est représentée par un sommet. Une route reliant deux villes est représentée par une arête. Sur chaque arête on note le nombre de kilomètres reliant les deux villes. Lorsque l'on munit ainsi chaque arête d'un **poids**, on parle d'un **graphe pondéré**. Étant donnés deux sommets A et B il s'agit de trouver un chemin de A à B , tel que la somme des poids est minimale. C'est un des problèmes les plus difficiles d'informatique théorique : lorsque le nombre n de sommets croît, le nombre de chemins reliant A et B explose et il n'est pas possible de les tester tous. Il s'agit d'un problème NP-difficile, c'est-à-dire qu'on pense qu'il n'existe pas d'algorithme donnant une réponse rapide (en temps polynomial).

Par contre il existe des algorithmes efficaces qui donnent une solution approchée, ce que vous utilisez tous les jours à l'aide du GPS de votre téléphone ou de votre voiture.

Exercice : trouver le plus court chemin reliant A à B .

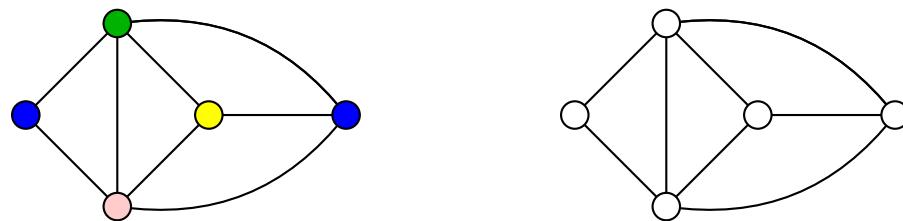


Il en est ainsi avec de nombreux problèmes sur les graphes : ils deviennent très durs à résoudre dès qu'on augmente le nombre de sommets.

Sudoku.

Le jeu du sodoku est en fait un problème de graphe. Tout d'abord expliquons ce qu'est le *problème de coloration* d'un graphe. Pour un graphe donné il s'agit de colorier chaque sommet, de sorte que deux sommets adjacents (c'est-à-dire reliés par une arête) soient de deux couleurs différentes. Bien sûr, on souhaite utiliser le moins de couleurs possibles.

Voici un exemple de graphe coloré avec 4 couleurs. Sauriez-vous trouver un coloriage avec seulement 3 couleurs ?



Revenons au Soduku. À une grille nous lui associons un graphe :

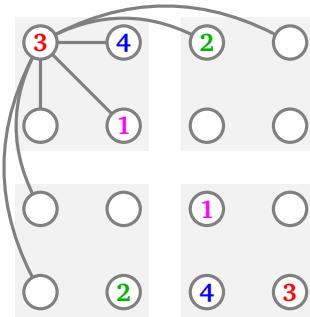
- chaque case est représentée par un sommet,
- si deux cases sont sur une même ligne ou une même colonne ou dans le même petit carré 3×3 alors on relie les sommets correspondants par une arête,
- chaque chiffre correspond à une couleur différente.

	4		5		1		9	
7			4		3			6
	3			6			8	
		1				6		
5	2						1	9
		4				8		
	9			3			7	
4			7		2			8
	1		9		8		6	

Grille de sudoku

Sur la figure de gauche ci-dessous vous avez un exemple d'une grille de sudoku 4×4 . Sur la droite le graphe a été partiellement dessiné : il y a bien les 16 sommets, mais il n'y a que les 7 arêtes partant du sommet en haut à gauche de représentées, il faudrait dessiner 7 arêtes par sommet. Chaque nombre est associé à une couleur, il s'agit donc de compléter la coloration du graphe (en tenant compte de toutes les arêtes).

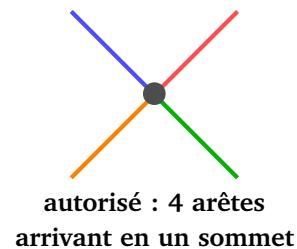
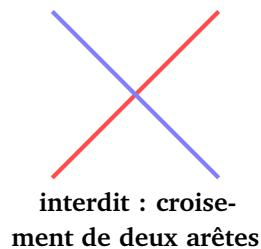
3	4	2	
	1		
		1	
	2	4	3



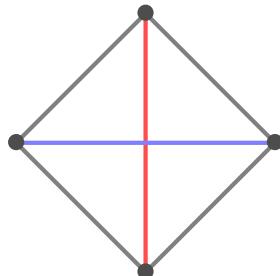
Le coloriage d'un graphe est aussi un problème NP-difficile.

1.3. Graphe planaire

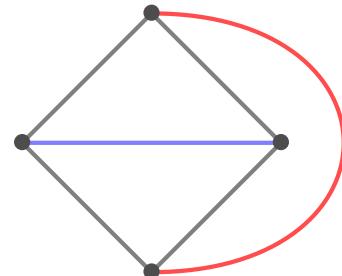
Un **graphe planaire** est un graphe que l'on peut dessiner dans le plan sans que les arêtes ne se croisent. Certains graphes ne sont pas planaires.



Il faut parfois changer la représentation du graphe pour savoir qu'il s'agit d'un graphe planaire.

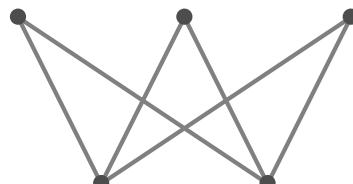


représentation non planaire



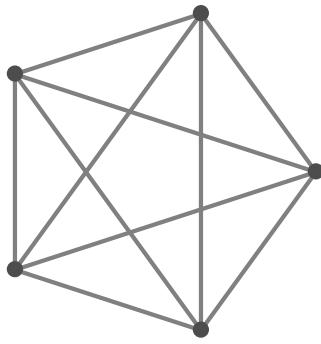
représentation planaire

Redessiner le graphe $K_{3,2}$ suivant afin justifier qu'il s'agit bien d'un graphe planaire :



Le graphe $K_{3,2}$

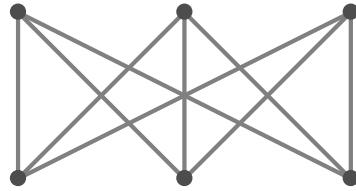
Peut-on relier 5 villes toutes entre elles par des routes qui ne se croisent pas ? Il s'agit donc de tracer un graphe complet à 5 sommets dans le plan. Ce n'est pas possible car le graphe K_5 suivant n'est pas planaire :



Il faut bien comprendre qu'il n'est pas possible de tracer les arêtes différemment afin qu'elles ne se croisent pas.

Trois villes veulent accéder à trois ressources. Est-il possible de relier chaque ville à toutes les ressources sans que les routes ne se croisent ?

La réponse est non car le graphe $K_{3,3}$ n'est pas non plus un graphe planaire.



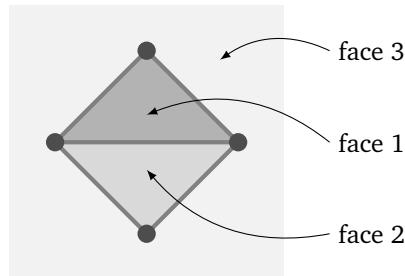
Le graphe $K_{3,3}$

1.4. Formule d'Euler

La formule d'Euler est une relation combinatoire pour les graphes planaires. Soit G un graphe planaire.

- On note S le nombre de sommets,
- on note A le nombre d'arêtes,
- on note F le nombre de faces.

Les **faces** sont les zones situées entre les arêtes. En termes mathématiques ce sont les composantes connexes du plan privé du graphe. Une des faces est infinie, les autres sont bornées.



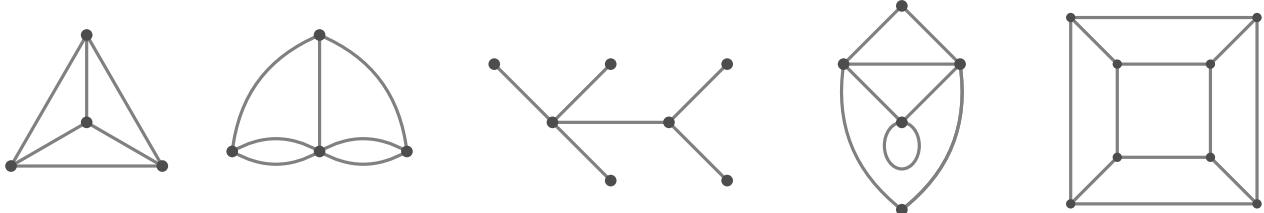
Théorème 1 (Formule d'Euler).

Soit G un graphe planaire connexe alors :

$$S - A + F = 2$$

Exercice.

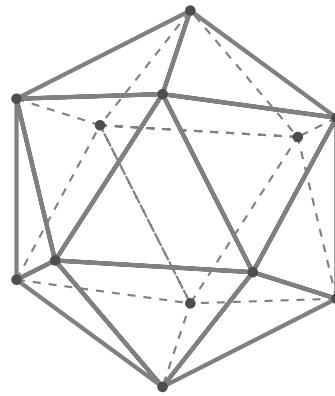
Vérifier la formule d'Euler sur les graphes suivants.



Voici une idée de la preuve qui se fait par récurrence sur le nombre d'arêtes : Tout d'abord pour deux sommets reliés par une arête la formule est vraie. Ensuite supposons que la formule soit vraie pour un graphe G . Premier cas : on rajoute une arête reliant un sommet existant à un nouveau sommet : on vérifie que la formule reste vraie pour ce nouveau graphe G' (on rajoute un sommet qui compense l'ajout d'une arête, le nombre de faces ne change pas). Second cas : on rajoute une arête reliant deux sommets existants, la formule est encore vraie car on rajoute une arête mais aussi une face.



La formule d'Euler est aussi valable pour les graphes tracés sur une sphère ou pour les solides de Platon. Vérifier la formule pour l'icosaèdre ci-dessous : c'est un dé à 20 faces, 30 arêtes et 12 sommets.

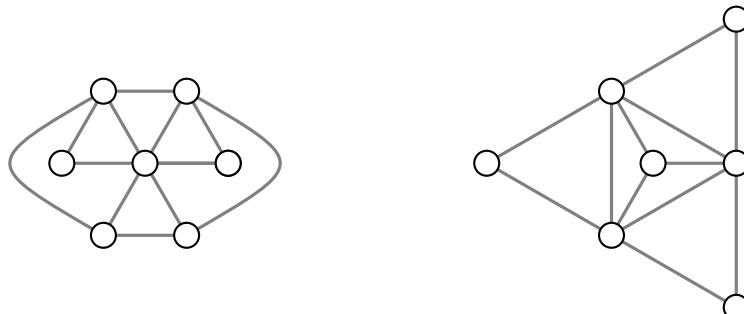


Combien faut-il de couleurs pour colorier un graphe du plan ? Le théorème suivant est devenu célèbre.

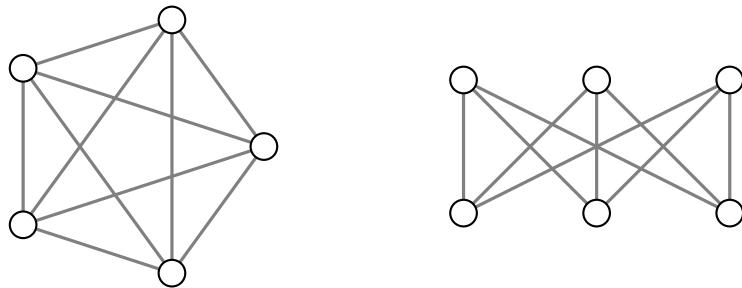
Théorème 2 (Théorème des 4 couleurs).

Un graphe planaire peut toujours être coloré avec 4 couleurs ou moins.

Colorez les graphes suivant avec 4 couleurs. Pouvez-vous faire moins ?



Attention pour les graphes non planaires ce n'est pas toujours possible. Par exemple le graphe K_5 ne peut pas être coloré avec 4 couleurs (pourquoi ?) par contre $K_{3,3}$ peut l'être (pourquoi ?).

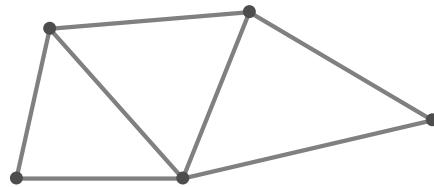


2. Triangulation

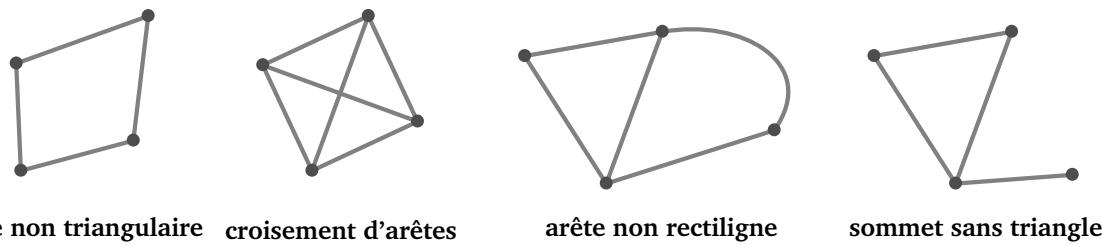
Une triangulation permet de découper le plan en éléments très simples.

2.1. Triangulation d'un polygone

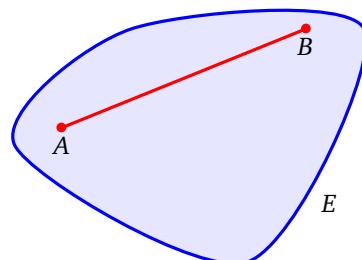
Une **triangulation**, c'est un graphe planaire dont toutes les faces sont des triangles (à l'exception de la face non bornée) et les arêtes sont des segments. Généralement un certain nombre de sommets ou d'arêtes sont imposés.



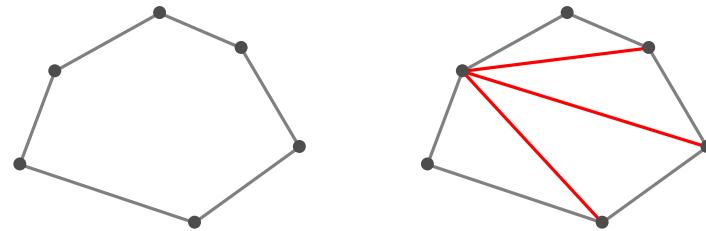
Voici des configurations interdites : faces qui ne sont pas des triangles, arêtes qui se croisent,...



On rappelle qu'un ensemble E est **convexe** si pour deux points quelconques A et B de E , alors le segment $[AB]$ est inclus dans E .

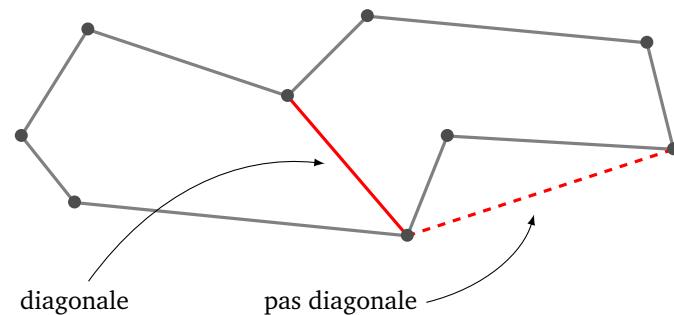


Il est très facile de trianguler un polygone convexe. Il s'agit de découper ce polygone en triangles (les sommets sont ceux du polygone, mais on doit rajouter des arêtes). On se donne donc un polygone convexe par une liste de sommets et d'arêtes. On choisit un sommet au hasard, et on trace les segments reliant ce point aux autres sommets qui ne sont pas ses voisins.



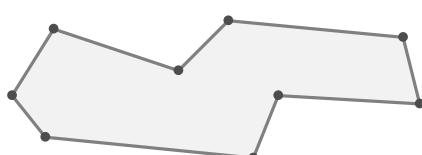
Considérons maintenant le cas d'un polygone non convexe. On suppose que ce polygone est *simple* (le bord est une suite d'arêtes qui ne se recoupent pas).

Une *diagonale* d'un polygone \mathcal{P} est un segment $[AB]$ reliant deux sommets et qui est inclus dans l'intérieur de \mathcal{P} . Nous admettons la propriété suivante : un polygone simple ayant plus de quatre sommets admet toujours une diagonale.

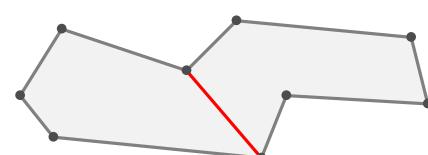


Un algorithme de triangulation procède alors par récurrence sur le nombre de sommets :

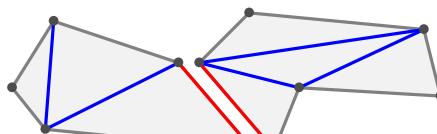
- si le polygone est un triangle alors il n'y a rien à faire,
- sinon, on coupe le polygone \mathcal{P} selon une diagonale $[AB]$. On obtient deux polygones \mathcal{P}_1 et \mathcal{P}_2 qui ont moins de sommets et donc que l'on sait trianguler (par hypothèse de récurrence).
- On recolle les triangulations de \mathcal{P}_1 et \mathcal{P}_2 le long de la diagonale $[AB]$ pour obtenir une triangulation de \mathcal{P} .



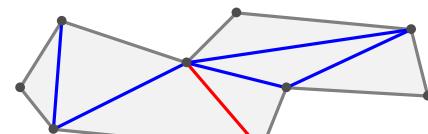
(a) Un polygone



(b) Une diagonale

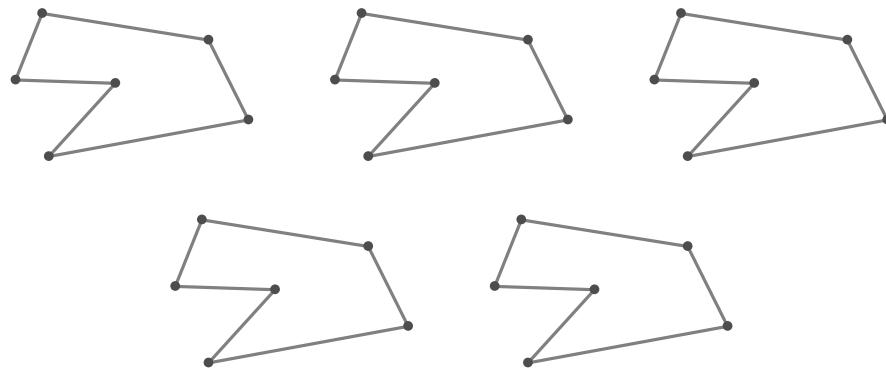


(c) Deux polygones triangulés



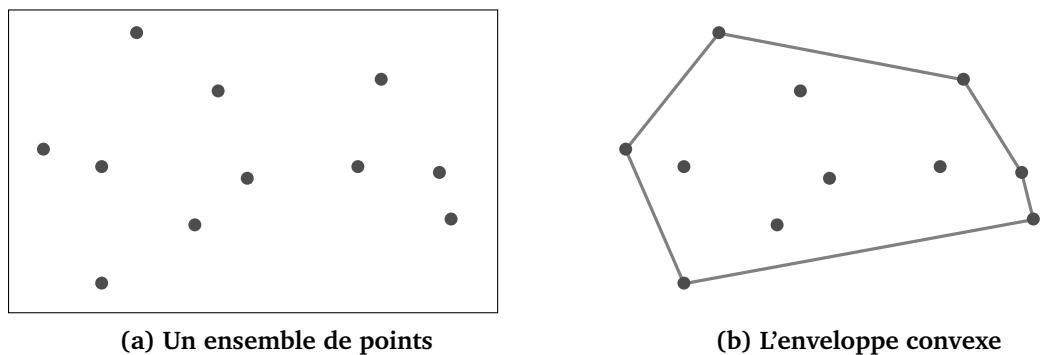
(d) Triangulation finale

Une triangulation n'est généralement pas unique. Par contre quelle que soit la triangulation d'un polygone à n sommets, le nombre de triangles est $n - 2$. Trouver toutes les triangulations de ce polygone.

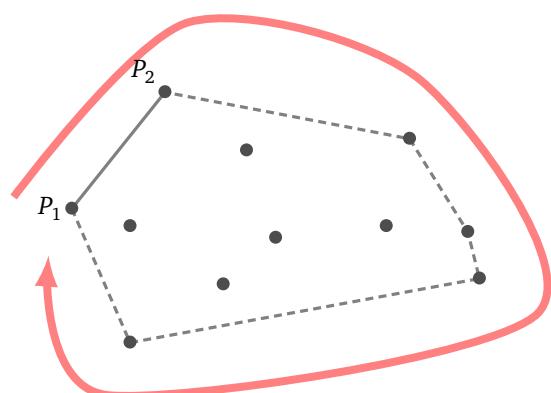


2.2. Enveloppe convexe

Nous partons maintenant d'un ensemble de points du plan. Nous commençons par déterminer son **enveloppe convexe**, c'est-à-dire le plus petit polygone convexe contenant tous ces points.



Nous allons calculer l'enveloppe convexe par l'algorithme de l'enroulement. Pour simplifier la description de l'algorithme nous faisons l'hypothèse de généricté suivante : il n'existe pas trois points alignés. En plus nous supposons que l'on a trié les points, le point P_1 étant le plus à gauche (et si plusieurs points avaient cette abscisse minimale, on prend P_1 le point le plus en bas). L'algorithme de l'enroulement consiste à entourer notre ensemble de points comme on emballerait un cadeau.



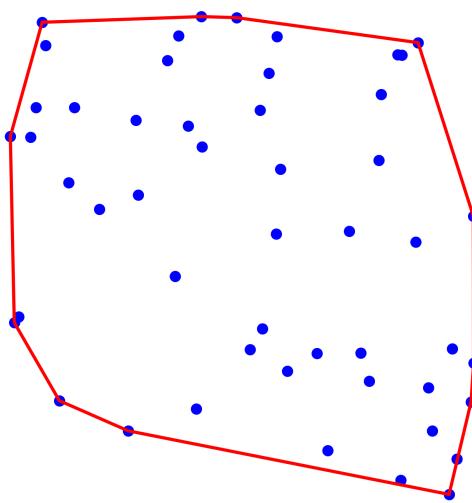
Algorithme.**Algorithme de l'enroulement.**

Entrée : un ensemble de points \mathcal{E} , dont P_1 est le point le plus à gauche.

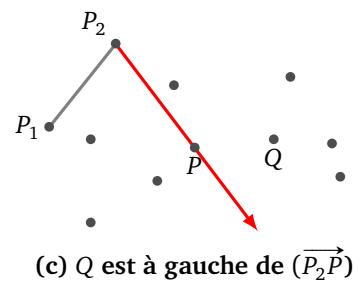
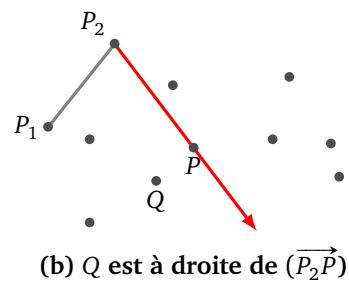
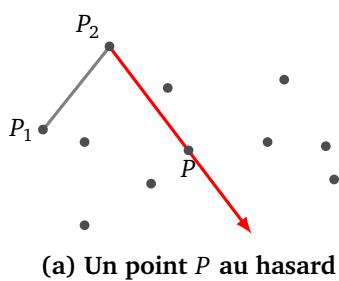
Sortie : une suite (P_1, P_2, \dots, P_m) de point formant l'enveloppe convexe de \mathcal{E} .

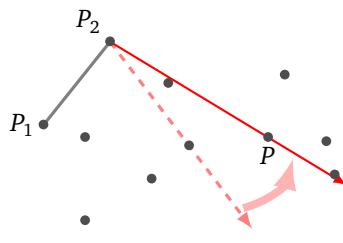
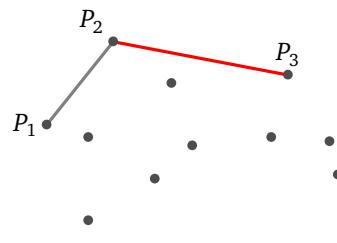
- On part du point le plus à gauche P_1 .
- On prend au hasard un autre point $P \in \mathcal{E}$.
- Pour tous les autres points $Q \in \mathcal{E}$:
 - Si Q est à gauche de la droite orientée $(\overrightarrow{P_1P})$ alors on fait $P = Q$.
- On pose $P_2 = P$.
- On recommence en partant de P_2 .
- ...
- On s'arrête lorsque l'on retombe sur P_1 .

Voici un exemple avec 50 points.



Voici les étapes pour trouver le second segment de l'enveloppe convexe : (a) on suppose que P_1 et P_2 sont déjà obtenus, on prend un point P au hasard parmi les points restants; (b) si un autre point Q est du côté droit de la droite orientée $(\overrightarrow{P_2P})$ alors on conserve P ; (c) et (d) si par contre Q est du côté gauche de $(\overrightarrow{P_2P})$ alors Q devient le nouveau point P ; (e) le dernier point P ainsi obtenu est le point P_3 qui donne le second segment de l'enveloppe convexe.



(d) Nouveau point P 

(e) Second segment

Comment décider si un point C est à droite ou à gauche de la droite orientée (\vec{AB}) ? C'est très simple :

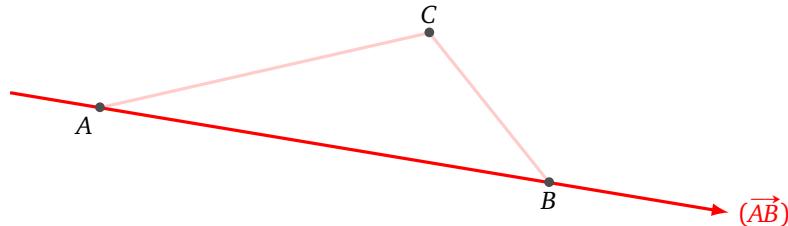
$$C \text{ est à gauche de } (\vec{AB})$$

$$\iff ABC \text{ est un triangle direct}$$

$$\iff \det(\vec{AB}, \vec{AC}) > 0$$

$$\iff \begin{vmatrix} x_B - x_A & x_C - x_A \\ y_B - y_A & y_C - y_A \end{vmatrix} > 0$$

$$\iff (x_B - x_A)(y_C - y_A) - (x_C - x_A)(y_B - y_A) > 0$$

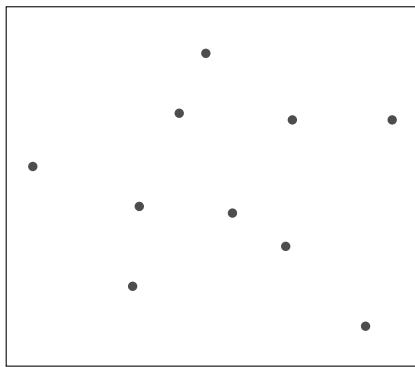


Complexité. Discutons un peu de la vitesse de notre algorithme. Bien sûr dans la pratique ce qui intéresse le programmeur et le joueur c'est de pouvoir afficher les images de sorte que l'animation soit fluide (par exemple 30 images par seconde). Pour faire abstraction du matériel, l'efficacité d'un algorithme se mesure via sa *complexité*, elle mesure le nombre d'opérations (ou la place en mémoire) nécessaire à l'algorithme en fonction de la taille n des données de départ. On donne généralement la complexité sous la forme $O(n)$ ou $O(n^2)$... où $O(n^k)$ signifie que le nombre d'opérations à effectuer est plus petit que Cn^k pour une certaine constante C (que l'on ne cherche pas à connaître en général). On peut aussi distinguer la complexité dans le pire des cas ou la complexité en moyenne.

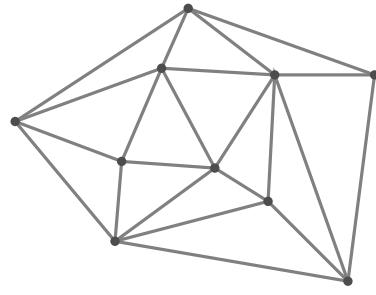
Notre algorithme a une complexité de $O(n^2)$: il faut de l'ordre de n^2 opérations (ici des calculs de déterminants 2×2). En effet il faut comparer chaque point P_i (au plus n choix) avec tous les autres points (encore n choix). Si n est petit (par exemple $n = 100$) alors trouver un meilleur algorithme n'est pas important, mais si n est beaucoup plus grand, ou si les calculs doivent se répéter 30 fois par seconde, alors trouver des algorithmes optimaux est important. Par exemple pour ce problème d'enveloppe convexe, les meilleurs algorithmes sont de complexité $O(n \log n)$ ce qui est une nette amélioration lorsque n est grand.

2.3. Triangulation d'un ensemble de points

Considérons maintenant le problème de la triangulation d'un ensemble \mathcal{E} de points. Il s'agit de trouver une triangulation telle que les points de \mathcal{E} soient exactement l'ensemble des sommets des triangles. On exige aussi que le bord de cette triangulation soit un polygone convexe.



(a) Un ensemble de points



(b) Une triangulation

Nous faisons de nouveau une hypothèse que les points sont en position générale : c'est-à-dire que trois points ne sont jamais alignés. On peut aussi supposer qu'il existe des points à l'intérieur de l'enveloppe convexe (sinon on renvoie à la triangulation d'un polygone convexe).

Algorithme.

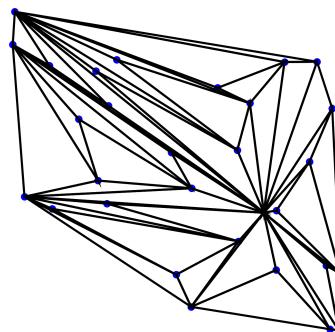
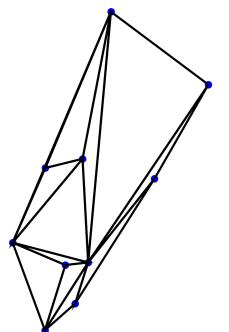
Algorithme de triangulation élémentaire.

Entrée : un ensemble de points \mathcal{E} .

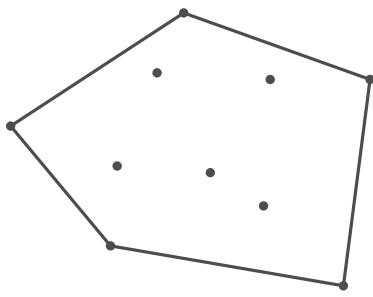
Sortie : une liste de triplets formant une triangulation.

- On calcule l'enveloppe convexe de \mathcal{E} .
- On choisit au hasard un point intérieur $P \in \mathcal{E}$. On trace les segments reliant P aux sommets de l'enveloppe convexe.
- Pour les autres points intérieurs $Q \in \mathcal{E}$:
 - on relie Q aux sommets du triangle qui le contient.

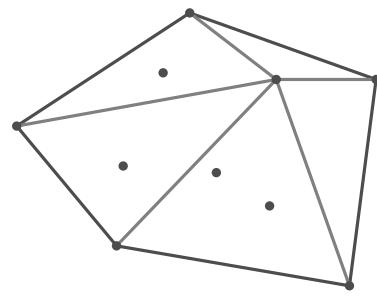
Ci-dessous à gauche une triangulation de 10 points et à droite de 30 points. On voit que via notre algorithme certains triangles sont très pointus et que beaucoup sont issus d'un même sommet (le point P de l'algorithme).



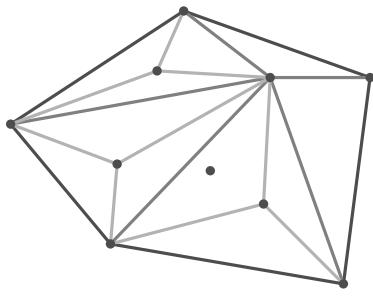
Voici les étapes de l'algorithme.



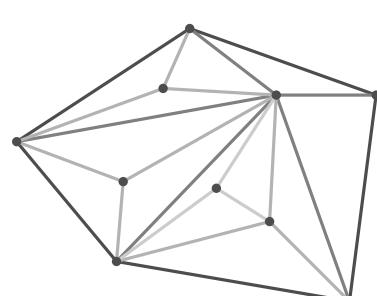
(a) Un ensemble de points et son enveloppe convexe



(b) Les triangles issus d'un point P



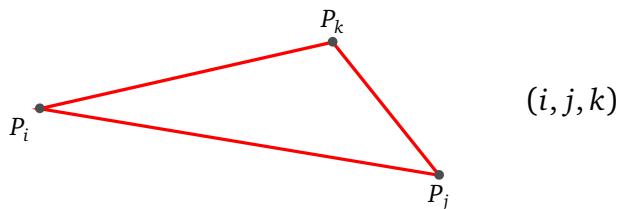
(c) On relie, un par un, un point aux sommets du triangle qui le contient



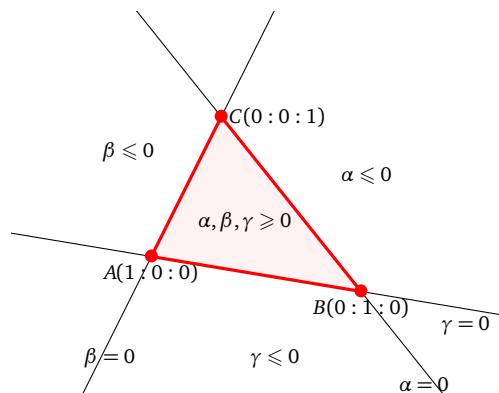
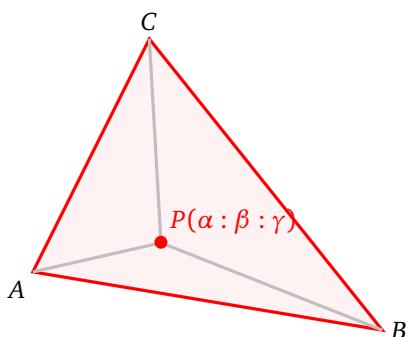
(d) On obtient une triangulation

Il y a plusieurs précisions à apporter.

Comment encoder une triangulation? On encode les points de \mathcal{E} par une liste (P_1, P_2, \dots, P_n) où chaque $P_i = (x_i, y_i)$ est un couple de coordonnées. On encode un triangle comme un triplet (i, j, k) correspondant au triangle $P_i P_j P_k$, on note \mathcal{T} l'ensemble de ces triplets. La triangulation est donc $(\mathcal{E}, \mathcal{T})$.



Comment déterminer si un point est à l'intérieur d'un triangle? Considérons un triangle ABC . Pour savoir si un point P est à l'intérieur du triangle on calcule ses coordonnées barycentriques $P(\alpha : \beta : \gamma)$ (avec $\alpha + \beta + \gamma = 1$). P est à l'intérieur du triangle si et seulement si $\alpha \geq 0$, $\beta \geq 0$ et $\gamma \geq 0$. On peut même préciser qu'il est sur un des côtés du triangle si $\alpha = 0$ ou $\beta = 0$ ou $\gamma = 0$.

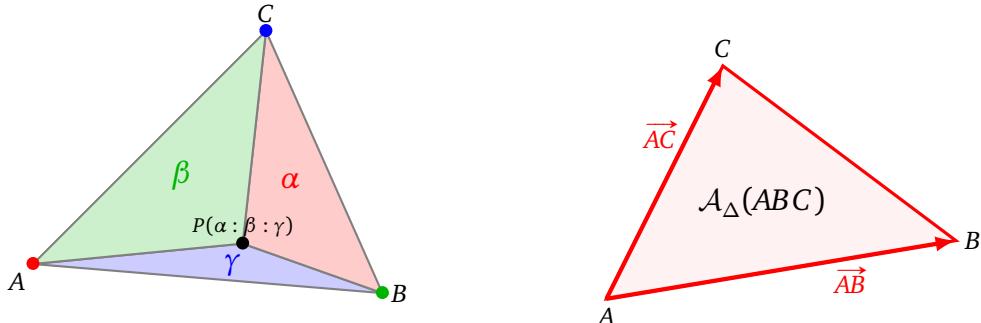


On renvoie au chapitre « Texture » pour les détails :

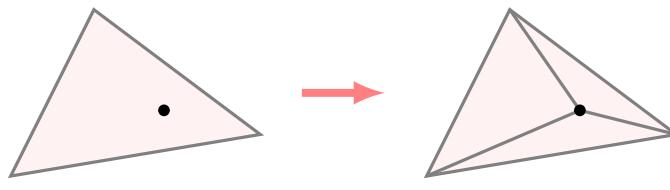
$$\alpha = \frac{A_{\Delta}(PBC)}{A_{\Delta}(ABC)} \quad \beta = \frac{A_{\Delta}(PCA)}{A_{\Delta}(ABC)} \quad \gamma = \frac{A_{\Delta}(PAB)}{A_{\Delta}(ABC)}$$

Et l'aire d'un triangle se calcule par la formule :

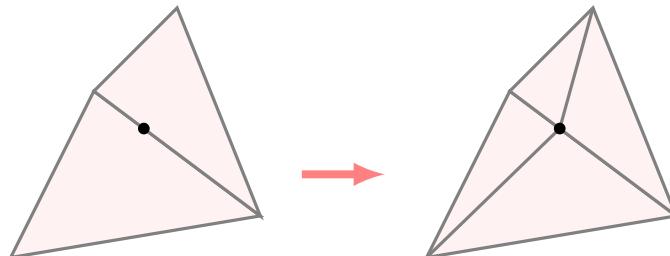
$$A_{\Delta}(ABC) = \frac{1}{2} |\det(\vec{AB}, \vec{AC})|$$



Que faire dans le cas où certains points sont alignés ? Dans la situation précédente on part d'un point P à l'intérieur d'un triangle, et on remplace ce triangle par trois nouveaux triangles. Si par contre P est sur un côté bordant deux triangles, alors on remplace ces triangles par quatre nouveaux triangles.



Cas sommet à l'intérieur



Cas sommet sur une arête

Une triangulation est-elle unique ? Non, il y a énormément de triangulations possibles. Par contre, quelle que soit la triangulation, le nombre de triangles reste le même.

Proposition 1.

Soit \mathcal{E} un ensemble de n points, dont l'enveloppe convexe contient b points. Le nombre de triangles d'une triangulation de \mathcal{E} est $2n - b - 2$. Le nombre d'arêtes est $3n - b - 3$.

Démonstration. Le nombre de sommets est $S = n$. Le nombre de faces est $F = t + 1$ où t est le nombre de triangles et où on n'oublie pas la face non bornée. Notons A le nombre d'arêtes. Si on considère une arête interne alors elle est le bord de deux triangles. Réciproquement un triangle est bordé par trois arêtes. Donc si on compte grossièrement on a deux arêtes pour trois triangles. Notre compte n'est pas tout fait juste car les arêtes du bord ne bordent qu'un seul triangle. La relation exacte est :

$$2A - b = 3t.$$

Appliquons maintenant la formule d'Euler $S - A + F = 2$. Dans cette formule on va substituer $S = n$, et de la relation $2A - b = 3t$ on tire $A = \frac{3t+b}{2}$, et enfin $F = t+1$. Ainsi la formule d'Euler devient $n - \frac{3t+b}{2} + t+1 = 2$. Ce qui donne la relation voulue : $t = 2n - b - 2$. Ce qui permet de calculer le nombre d'arêtes $A = 3n - b - 3$. \square

3. Triangulation de Delaunay

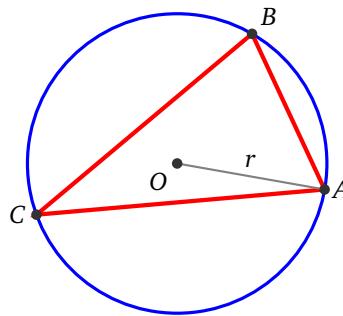
3.1. Cercle circonscrit

Proposition 2.

Le cercle circonscrit \mathcal{C} d'un triangle ABC a pour centre O de coordonnées (x_O, y_O) où :

$$x_O = \frac{1}{2\Delta} \begin{vmatrix} x_A^2 + y_A^2 & y_A & 1 \\ x_B^2 + y_B^2 & y_B & 1 \\ x_C^2 + y_C^2 & y_C & 1 \end{vmatrix} \quad \text{et} \quad y_O = -\frac{1}{2\Delta} \begin{vmatrix} x_A^2 + y_A^2 & x_A & 1 \\ x_B^2 + y_B^2 & x_B & 1 \\ x_C^2 + y_C^2 & x_C & 1 \end{vmatrix} \quad \text{avec} \quad \Delta = \begin{vmatrix} x_A & y_A & 1 \\ x_B & y_B & 1 \\ x_C & y_C & 1 \end{vmatrix}$$

et a pour rayon $r = OA = \sqrt{(x_A - x_O)^2 + (y_A - y_O)^2}$.

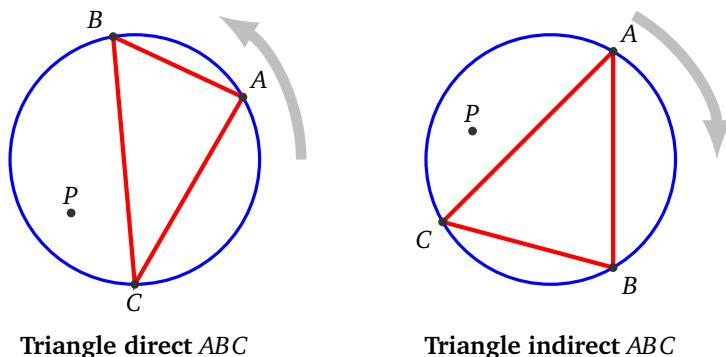


Proposition 3.

Soit ABC un triangle direct. Un point $P(x, y)$ est sur le bord ou à l'intérieur du cercle \mathcal{C} si et seulement si :

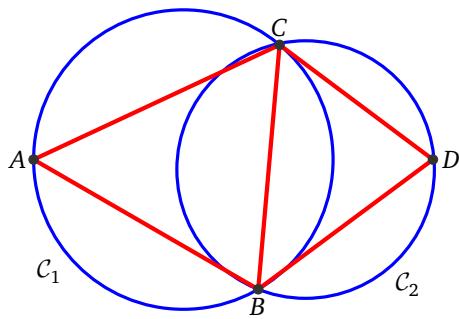
$$\begin{vmatrix} x^2 + y^2 & x & y & 1 \\ x_A^2 + y_A^2 & x_A & y_A & 1 \\ x_B^2 + y_B^2 & x_B & y_B & 1 \\ x_C^2 + y_C^2 & x_C & y_C & 1 \end{vmatrix} \leq 0$$

Dans le cas d'un triangle indirect, le point P est à l'intérieur de \mathcal{C} si le déterminant est positif.



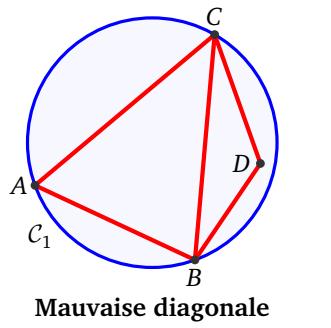
Considérons un quadrilatère convexe de sommets A, B, C, D comme sur la figure ci-dessous.

- On dit que la diagonale $[BC]$ est **bonne** si le cercle \mathcal{C}_1 circonscrit à ABC ne contient pas le point D dans son intérieur et si le cercle \mathcal{C}_2 circonscrit à BCD ne contient pas le point A dans son intérieur.

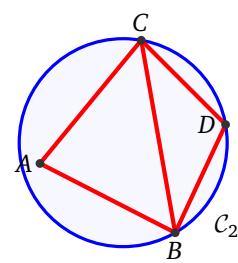


Bonne diagonale

- Dans le cas contraire on dit que la diagonale $[BC]$ est **mauvaise** : soit le cercle \mathcal{C}_1 circonscrit à ABC contient le point D dans son intérieur ou bien le cercle \mathcal{C}_2 circonscrit à BCD contient le point A dans son intérieur.



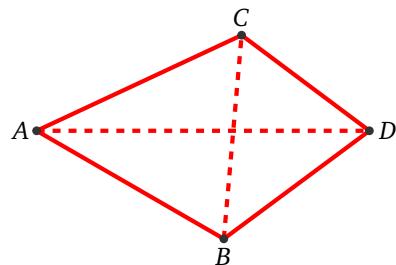
Mauvaise diagonale



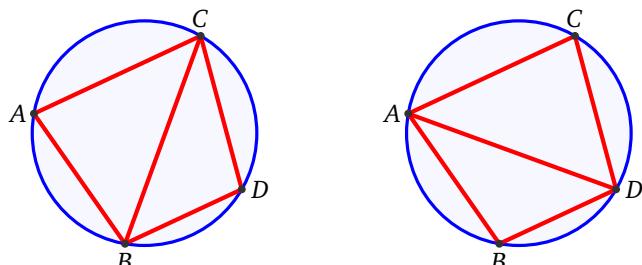
Mauvaise diagonale

Proposition 4 (Mauvaise arête et basculement).

Considérons un quadrilatère convexe de sommets A, B, C, D . Soit la diagonale $[BC]$ est bonne, soit la diagonale $[AD]$ est bonne.



L'une des deux diagonales est donc une bonne diagonale. Autrement dit, les deux diagonales ne peuvent être mauvaises simultanément. Le seul cas où les deux diagonales sont bonnes en même temps est lorsque les points A, B, C, D sont cycliques.



Cas cocyclique : les deux diagonales sont bonnes

3.2. Triangulation de Delaunay

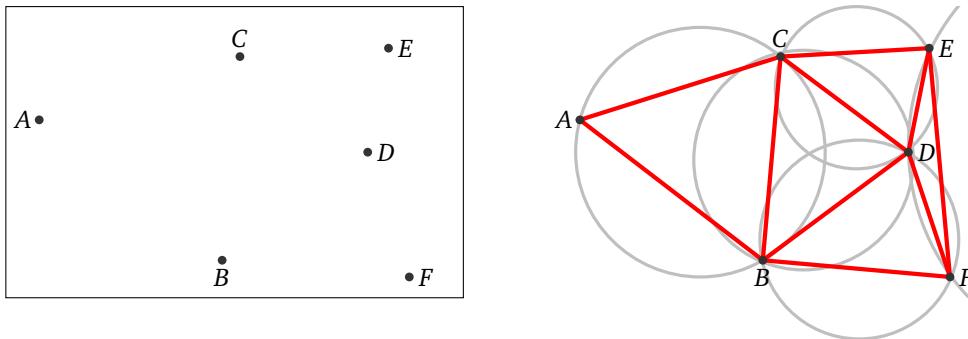
Définition.

Soit \mathcal{E} un ensemble de points. On dit qu'une triangulation \mathcal{T} est une **triangulation de Delaunay** de \mathcal{E} si pour tout triangle T de \mathcal{T} , le cercle circonscrit à T ne contient aucun point de \mathcal{E} dans son intérieur strict.

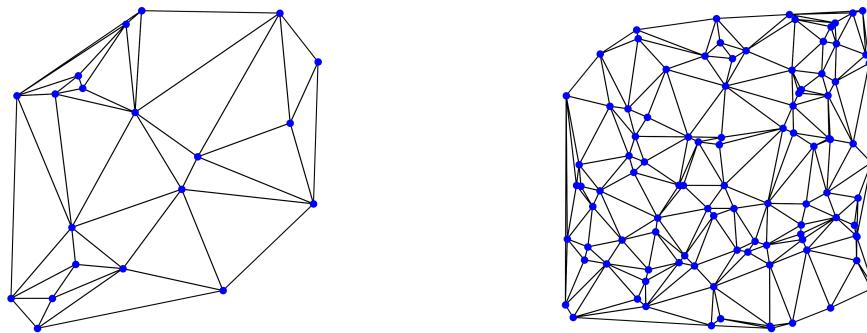
Le premier point fort d'une triangulation de Delaunay est qu'il n'y en a (presque) qu'une seule.

Théorème 3.

Pour n'importe quel ensemble fini de points, il existe une triangulation de Delaunay. De plus cette triangulation est unique, à l'exception des sommets cocycliques où un basculement fournit deux possibilités.

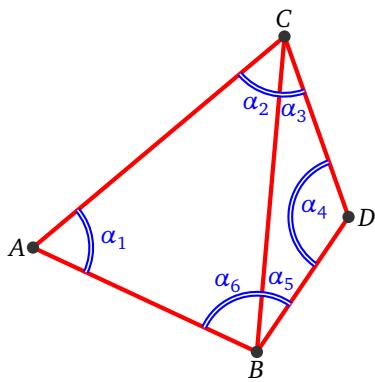


Ci-dessous une triangulation de Delaunay de 20 points (à gauche) et de 100 points (à droite).

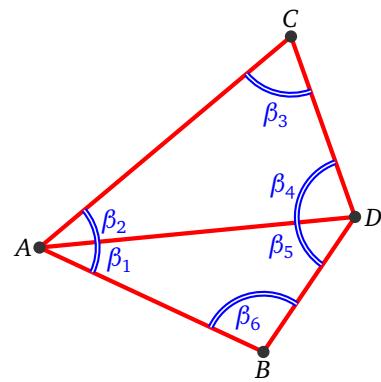


Pourquoi une triangulation de Delaunay serait-elle meilleure qu'une autre ? La triangulation de Delaunay est celle qui évite le mieux les triangles ayant des angles très aigus, c'est-à-dire les petits angles où le triangle est pointu. Bien sûr on ne peut pas toujours éviter les triangles pointus mais la triangulation de Delaunay est celle qui le fait le mieux (elle réalise un minimum global d'une certaine fonction évaluant tous les angles de tous les triangles).

Le basculement s'explique bien avec cette notion d'angle, lorsque l'on bascule une arête d'un quadrilatère, la seule chose qui change ce sont les 6 angles. Lorsque l'on bascule une mauvaise arête, alors l'angle le plus petit avant basculement est plus petit que l'angle le plus petit après : $\min_{i=1,\dots,6} \alpha_i < \min_{i=1,\dots,6} \beta_i$.



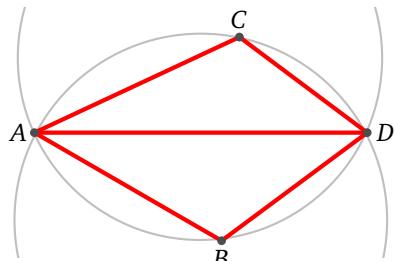
Mauvaise diagonale



Bonne diagonale

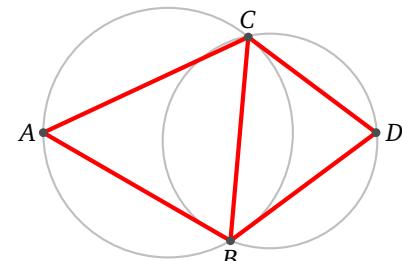
3.3. Algorithme par basculement

Rappelons que dans un quadrilatère, si l'arête diagonale est mauvaise, on peut toujours la transformer en une bonne arête par **basculement**. Cela va nous permettre de calculer une triangulation de Delaunay à partir d'une triangulation quelconque.



Mauvaise diagonale

basculement



Bonne diagonale

Algorithme.

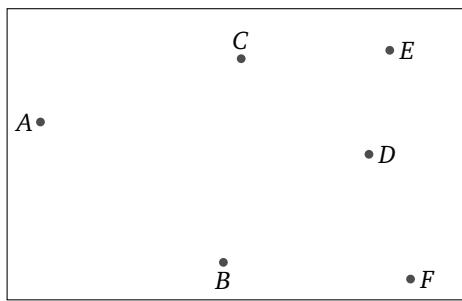
Algorithme de basculement.

Entrée : un ensemble de points \mathcal{E} muni d'une triangulation \mathcal{T} .

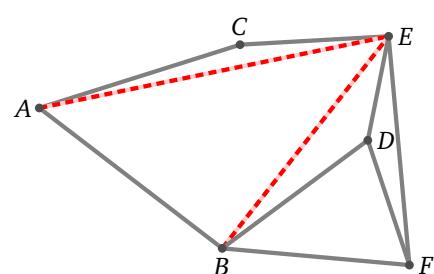
Sortie : la triangulation de Delaunay de \mathcal{E} .

- Tant qu'il existe un quadrilatère avec une mauvaise diagonale :
 - basculer cette arête en une bonne diagonale.

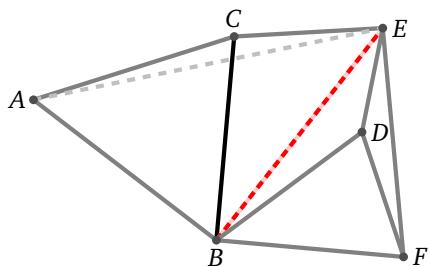
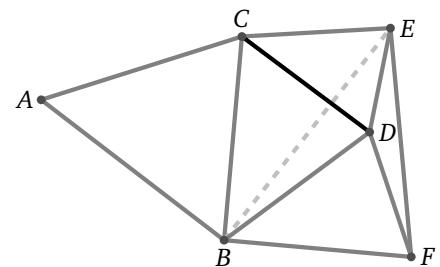
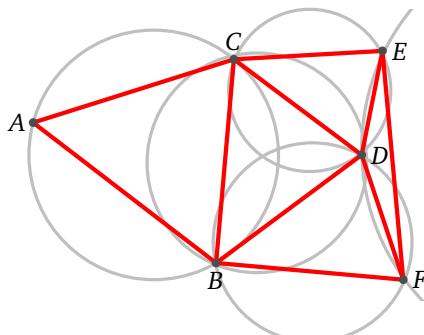
L'algorithme est donc très simple. Voici un exemple :



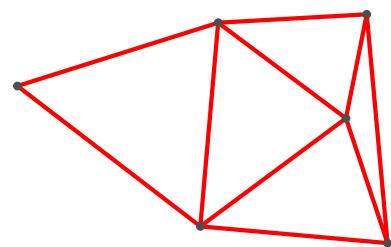
Un ensemble de points



Une triangulation et des mauvaises diagonales

Basculement de $[AE]$ vers $[BC]$ Basculement de $[BE]$ vers $[CD]$ 

Cercles circonscrits



Triangulation de Delaunay

Reprenez cet ensemble de points, dessinez une autre triangulation, appliquez l'algorithme : vous devez obtenir la même triangulation finale !

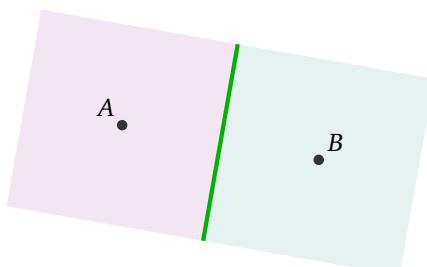
Il y a quand même des subtilités : en cours d'exécution de l'algorithme le caractère bon ou mauvais d'une diagonale peut changer, en effet les triangles (donc les quadrilatères) changent au fil de l'algorithme, il ne s'agit donc pas de calculer toutes les mauvaises diagonales au départ et de les basculer. L'algorithme cherche une mauvaise diagonale, la bascule, puis cherche une mauvaise diagonale dans la nouvelle triangulation... Vu la remarque précédente il n'est pas évident du tout que l'algorithme se termine. Une arête que l'on vient de basculer en une bonne diagonale, peut-elle se retrouver plus tard une mauvaise diagonale ? Heureusement non, ce qui fait que l'algorithme termine toujours par une triangulation sans mauvaises diagonale, c'est-à-dire une triangulation de Delaunay. Cet algorithme élémentaire n'est pas très rapide, sa complexité est $O(n^2)$ où n est le nombre de sommets. Il existe des algorithmes de triangulation de Delaunay en $O(n)$.

4. Cellules de Voronoï

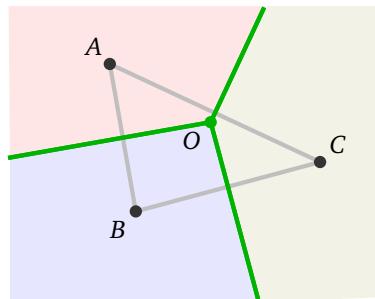
4.1. La guerre des princesses

Des princesses habitent un pays, chacune dans son château. Pour se partager le territoire en évitant de se faire la guerre elles décident de la règle suivante : « Chacune est propriétaire du territoire plus proche de son château que des autres châteaux. »

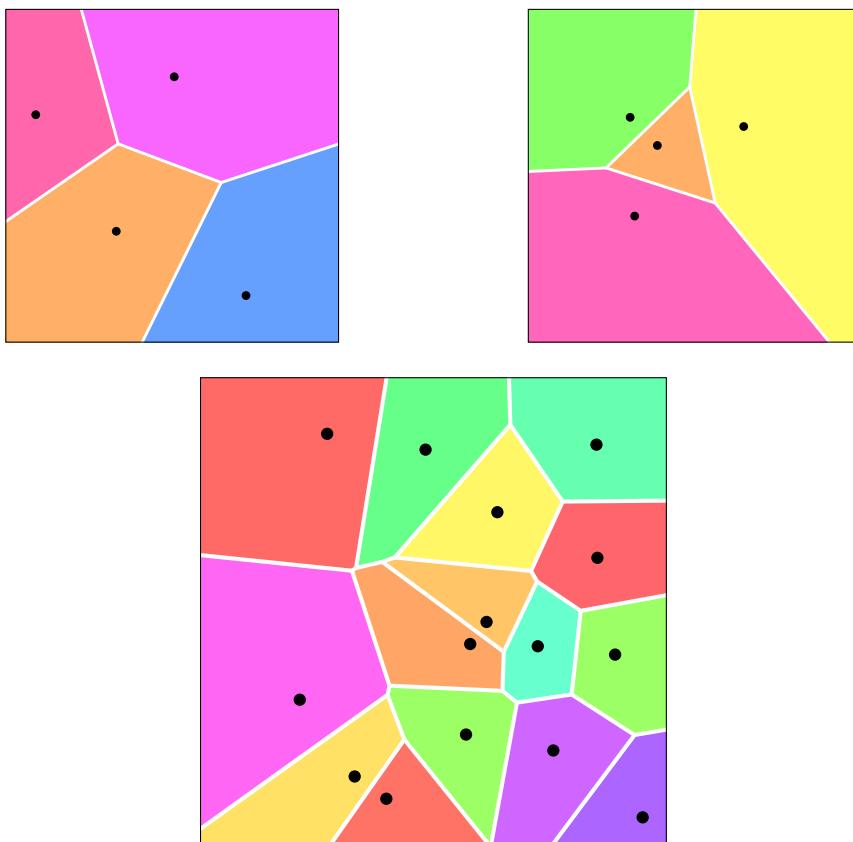
S'il y a seulement deux princesses A et B alors chacune contrôle un demi-plan. La séparation se fait selon la médiatrice $[AB]$ des deux châteaux.



S'il y a trois princesses A, B, C c'est déjà plus intéressant. On dessine les médiatrices du triangle ABC et leur point d'intersection O (qui est aussi le centre du cercle circonscrit). Les zones sont délimitées par les « demi-médiatrices » issues de O .



La zone contrôlée par une princesse s'appelle une **cellule de Voronoï**. S'il y a 4 princesses ou plus alors plusieurs types de configurations sont possibles. Certaines cellules de Voronoï peuvent être bornées, d'autres non.



Les applications sont nombreuses, par exemple les cellules vivantes se développent jusqu'à rencontrer les cellules voisines, de même un sol asséché se craquelle en cellules de Voronoï, enfin votre téléphone se connecte à l'antenne relais la plus proche c'est-à-dire lorsque vous êtes dans sa cellule de Voronoï.

4.2. Dual de Delaunay

Tracer les cellules de Voronoï d'un ensemble de points c'est très simple à partir de la triangulation de Delaunay. Les sommets des cellules sont les centres des cercles circonscrits à chaque triangle de la triangulation. Les bords des cellules sont les portions des médiatrices de ces triangles reliant deux sommets voisins. Les cellules de Voronoï correspondent au *graphe dual* de la triangulation de Delaunay.

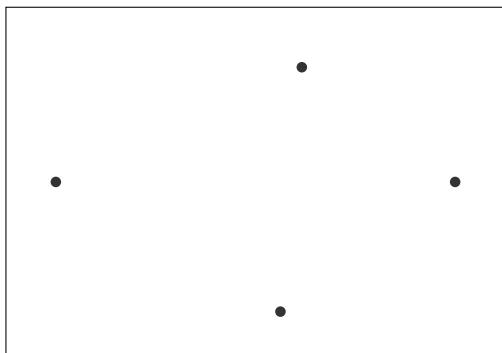
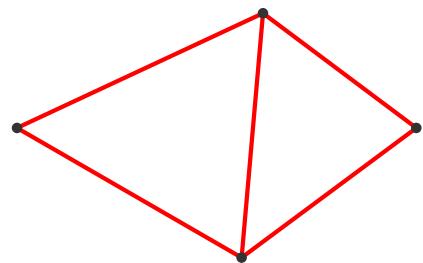
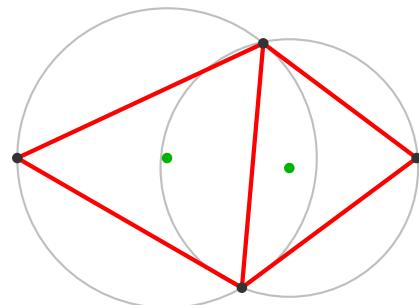
Algorithme.**Algorithme des cellules de Voronoï.**

Entrée : un ensemble de points \mathcal{E} muni d'une triangulation \mathcal{T} .

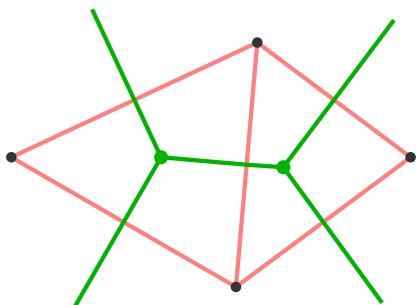
Sortie : le graphe \mathcal{V} de Voronoï de \mathcal{E} .

- Calculer la triangulation \mathcal{T} de Delaunay de \mathcal{E} .
- Les sommets de \mathcal{V} sont les centres de cercles circonscrits des triangles de \mathcal{T} .
- Les arêtes de \mathcal{V} sont les portions de médiatrices des côtés des triangles de \mathcal{T} joignant deux sommets voisins (ou infinies s'il n'y a pas de voisin).

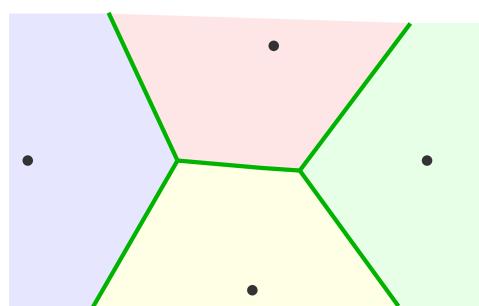
Voici les étapes de l'algorithme.

(a) Un ensemble de points \mathcal{E} (b) La triangulation de Delaunay \mathcal{T} 

(c) Les cercles circonscrits et leurs centres

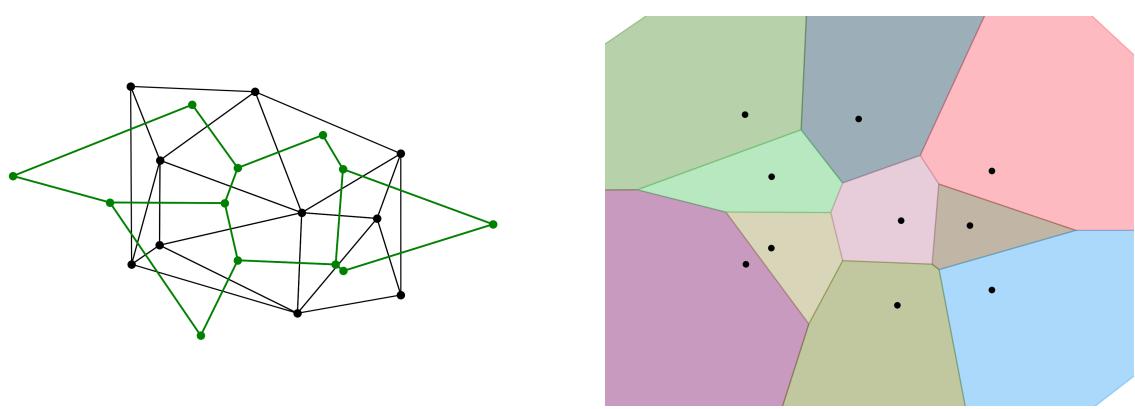


(d) Les portions de médiatrices

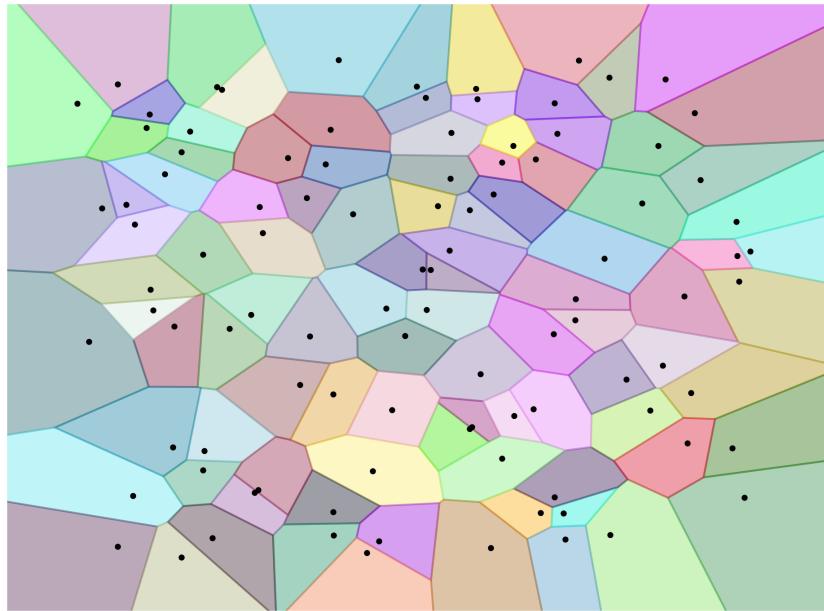


(e) Les cellules de Voronoï

Voici un exemple avec 10 points. Sur la figure de gauche on a dessiné la triangulation de Delaunay (en noir) et sa triangulation duale (en vert) qui délimite les cellules de Voronoï (les demi-médiatrices infinies ne sont pas tracées). Sur la figure de droite les cellules de Voronoï de ces mêmes points.

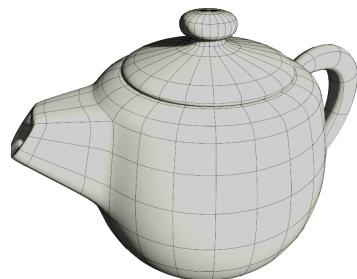


Voici un exemple avec 100 points.



Maillage

Cette fois nous découpons un objet de l'espace en figures géométriques simples.



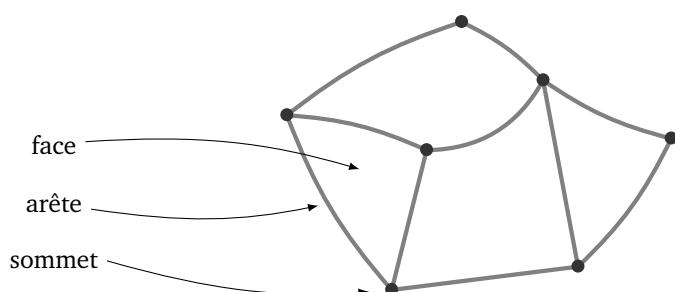
1. Généralités

1.1. Une définition

Un **maillage** d'une surface E est un ensemble (S, A, F) où :

- S sont des **sommets** (ici des points de l'espace),
- A sont des **arêtes** (ici des portions de courbes),
- F sont des **faces** (ici des portions de surfaces),

de telle sorte que l'union de toutes les faces soit égale à la surface E de départ.



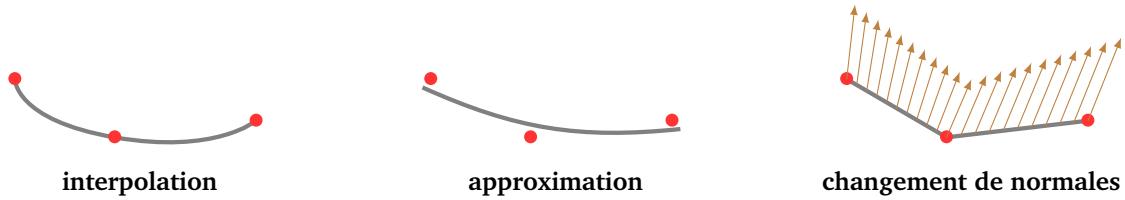
On impose en plus des conditions reliant sommets/arêtes/faces :

- Le bord d'une arête est composé de deux sommets distincts. Tout sommet est dans le bord d'une arête. Deux arêtes ne peuvent s'intersecter qu'en un sommet.
- Le bord d'une face est formé d'arêtes distinctes. Toute arête est dans le bord d'une face. L'intersection de deux faces est une arête ou un sommet (ou rien).

Partant d'une surface originale E , un maillage s'obtient souvent par discréétisation de la surface, comme pour les triangulations (voir le chapitre « Triangulation »). Cependant ici les objets sont des objets de l'espace et les faces ne sont pas nécessairement des triangles. Par exemple une arête peut être une courbe de l'espace et une face n'est pas nécessairement plate.

On peut aussi considérer un maillage comme un objet combinatoire, c'est-à-dire une fois donnés les sommets S , une arête est juste un couple de sommets (sans description de la courbe les reliant) et une face est une liste de sommets. Il y a alors plusieurs façons de reconstruire une surface correspondant à ce maillage :

- **interpolation** : on construit une courbe ou une surface qui passe par les sommets imposés,
- **approximation** : on construit une courbe ou une surface qui approche au mieux (selon certains critères) les sommets mais qui ne passe pas nécessairement par ces sommets,
- **changement de normales** : on ne modifie pas le maillage mais la manière dont l'éclairage est calculé en modifiant les vecteurs normaux à chaque face (voir le chapitre « Texture »).



1.2. Codage

Revenons sur la combinatoire (S, A, F) d'un maillage et comment on pourrait l'encoder dans un ordinateur :

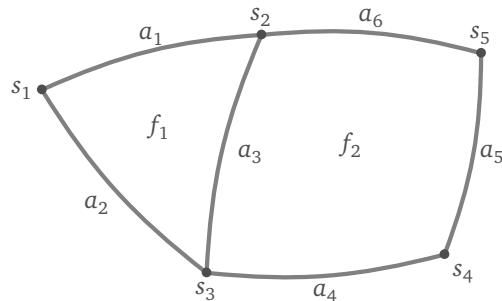
- L'ensemble des sommets S est une liste numérotée de points s_i , $i = 1, \dots, n$. Chaque sommet s_i est déterminé par des coordonnées $(x_i, y_i, z_i) \in \mathbb{R}^3$.

Données optionnelles. À chaque sommet on associe la liste des arêtes issues de ce sommet.
- L'ensemble des arêtes A est une liste numérotée d'arêtes $a_j = (i_1, i_2)$ où i_1 et i_2 sont les numéros des sommets s_{i_1} et s_{i_2} aux bords de l'arête.

Données optionnelles. À chaque arête on associe la liste des faces touchant cette arête.
- L'ensemble des faces F est une liste numérotée de faces f_k définies par une liste (i_1, i_2, \dots, i_m) de (numéros de) sommets.

Données optionnelles. À chaque face on associe la liste des arêtes du bord.

Quelles sont les données du maillage dessiné ci-dessous ?



- $S = \{s_1, \dots, s_5\}$, avec $s_i = (x_i, y_i, z_i)$, $i = 1, \dots, 5$.
- Les arêtes sont $A = \{a_1, \dots, a_6\}$, avec $a_1 = \{1, 2\}$ (car d'extrémités s_1 et s_2), $a_2 = \{1, 3\}, \dots$
- Les faces sont $F = \{f_1, f_2\}$, avec $f_1 = \{1, 2, 3\}$ (car déterminée par les sommets s_1, s_2, s_3) et $f_2 = \{2, 3, 4, 5\}$.

Les données optionnelles peuvent être obtenues à partir des données initiales mais nécessitent un parcours dans ces données initiales.

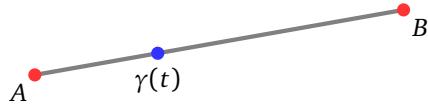
1.3. Quelques types d'arêtes

Pour deux sommets A et B il existe de nombreuses façons de les relier. Voyons trois méthodes.

Arête rectiligne/lerp.

$$\gamma(t) = (1-t)A + tB \quad t \in [0, 1]$$

C'est l'interpolation linéaire classique (*lerp*) qui dessine un arête rectiligne. Il faut comprendre l'addition de points comme l'addition de vecteurs $A = \begin{pmatrix} x_A \\ y_A \end{pmatrix}$, $B = \begin{pmatrix} x_B \\ y_B \end{pmatrix}$ (dans le plan) ou $A = \begin{pmatrix} x_A \\ y_A \\ z_A \end{pmatrix}$, $B = \begin{pmatrix} x_B \\ y_B \\ z_B \end{pmatrix}$ (dans l'espace).

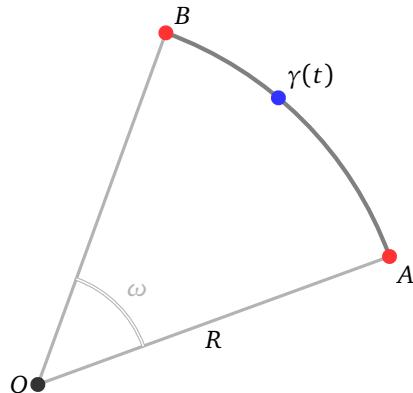


Arête circulaire/slerp. Considérons deux points A et B situés à une distance R d'un point O . Alors l'arc du cercle de centre O et de rayon R commençant à A et finissant à B est donné par :

$$\boxed{\gamma(t) = R \frac{\sin((1-t)\omega)}{\sin \omega} A + R \frac{\sin(t\omega)}{\sin \omega} B \quad t \in [0, 1]}$$

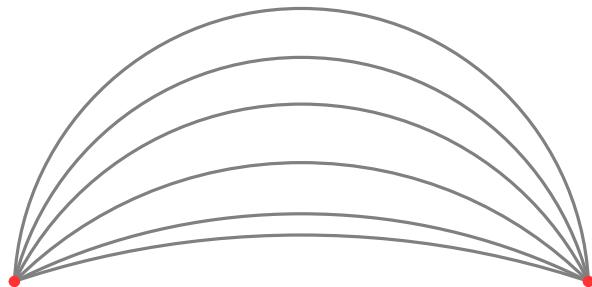
où ω est l'angle formé par les vecteurs \vec{OA} et \vec{OB} et est donné par la formule :

$$\vec{OA} \cdot \vec{OB} = R^2 \cos(\omega).$$

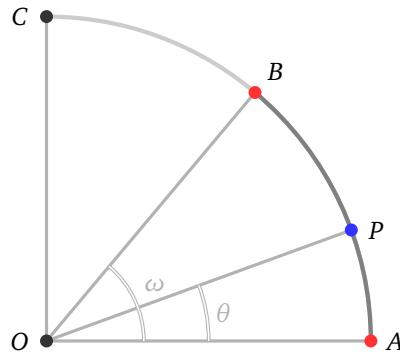


Par exemple en $t = \frac{1}{2}$, on obtient le milieu de l'arc ; avec $t = \frac{1}{3}$ et $t = \frac{2}{3}$ on coupe l'arc en trois parties égales...

Pour différentes valeurs du rayon on obtient des arcs plus ou moins courbés.



Preuve : On peut se placer dans le plan (Oxy) et supposer que A a pour coordonnées $(1, 0)$. Notons C le point de coordonnées $(0, 1)$. Dans ce repère les coordonnées d'un point du cercle centré en $O(0, 0)$ et de rayon 1 sont $(\cos \theta, \sin \theta)$. Ainsi un point P du cercle s'écrit $P = \cos(\theta)A + \sin(\theta)C$ (c'est-à-dire $\vec{OP} = \cos(\theta)\vec{OA} + \sin(\theta)\vec{OC}$). En particulier $B = \cos(\omega)A + \sin(\omega)C$.



Ainsi $C = -\frac{\cos \omega}{\sin \omega}A + \frac{1}{\sin \omega}B$. On obtient alors :

$$P = \frac{\sin \omega \cos \theta - \sin \theta \cos \omega}{\sin \omega}A + \frac{\sin \theta}{\sin \omega}B = \frac{\sin(\omega - \theta)}{\sin \omega}A + \frac{\sin \theta}{\sin \omega}B.$$

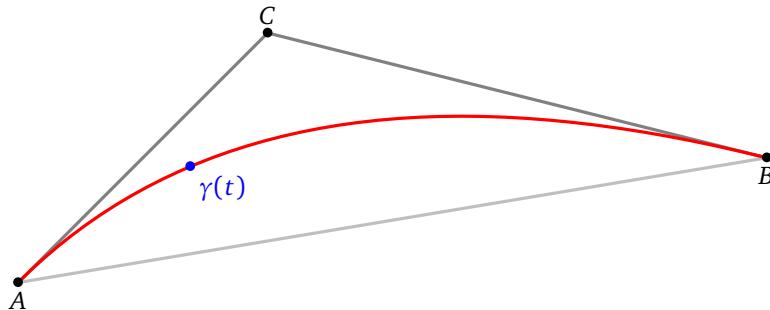
On pose $\theta = \omega t$ où $t \in [0, 1]$ afin d'obtenir la paramétrisation de l'arc de cercle entre A (en $t = 0$) et B (en $t = 1$) :

$$P = \frac{\sin((1-t)\omega)}{\sin \omega}A + \frac{\sin(t\omega)}{\sin \omega}B.$$

Arête quadratique. Ces arêtes sont données par des équations de degré 2. Autrement dit, ce sont des courbes de Bézier avec un seul point de contrôle. La **courbe quadratique de Bézier** de A à B ayant le point de contrôle C est la courbe paramétrée :

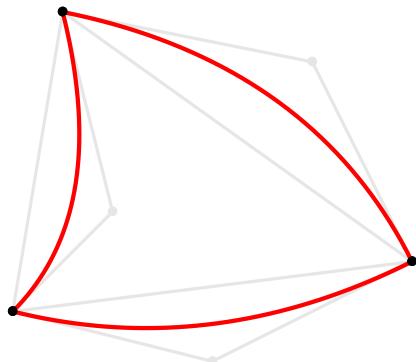
$$\gamma(t) = (1-t)^2A + 2t(1-t)C + t^2B \quad t \in [0, 1]$$

La courbe démarre en $\gamma(0) = A$ et termine en $\gamma(1) = B$, elle ne passe pas par C . Par contre la courbe est tangente à \overrightarrow{AC} en A et tangente à \overrightarrow{BC} en B .



Les courbes de Bézier générales seront étudiées dans le chapitre « Approximation et interpolation ».

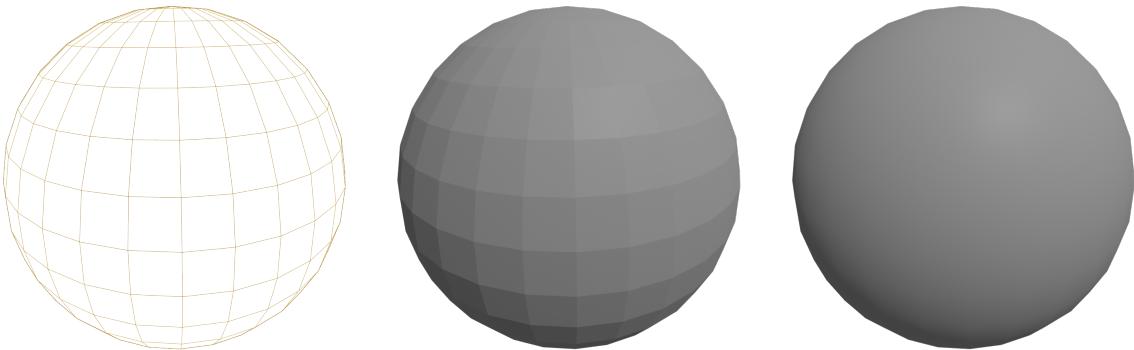
Un **triangle de Bézier quadratique** est défini par 3 sommets et 3 points de contrôle.



2. Vecteur normal

On rappelle que l'éclairage d'un objet est calculé à partir des vecteurs normaux à la surface (voir les chapitres « Lumière » et « Texture »). Changer les vecteurs normaux permet de changer l'apparence de l'objet sans toucher à la géométrie sous-jacente.

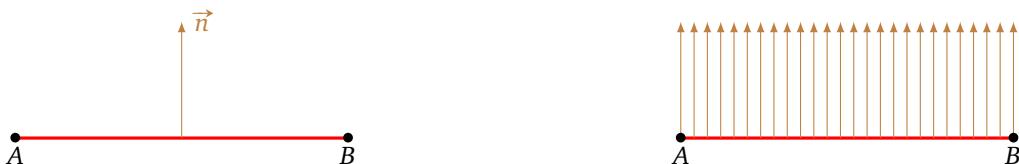
Ci-dessous de gauche à droite : le maillage d'une sphère, le rendu plat et le rendu après lissage des vecteurs normaux (sans changer le maillage).



2.1. Normales aux arêtes

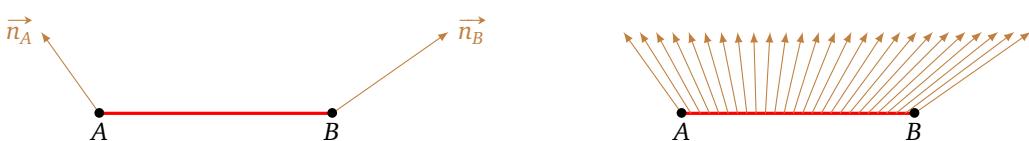
Nous commençons par le cas d'une arête entre deux points A et B dans le plan.

Une seule normale. Quel que soit le point P du segment $[AB]$, le vecteur normal est toujours le même (à un facteur multiplicatif près). Cela correspond à un éclairage « plat ».

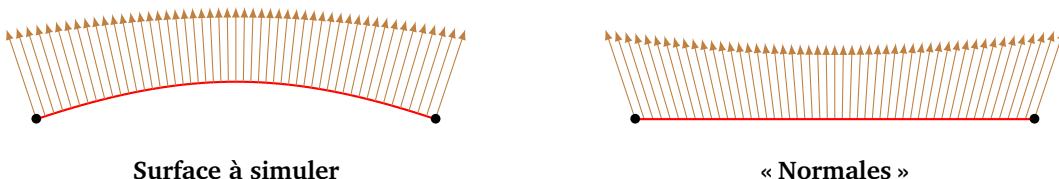


Une normale à chaque extrémité. Considérons le même segment $[AB]$ et supposons que l'on nous donne un vecteur \vec{n}_A en A et un vecteur \vec{n}_B en B . On en déduit un vecteur en n'importe quel point du segment. Par interpolation linéaire, un point P du segment s'écrit $P = (1 - t)A + tB$ et en ce point on définit le vecteur :

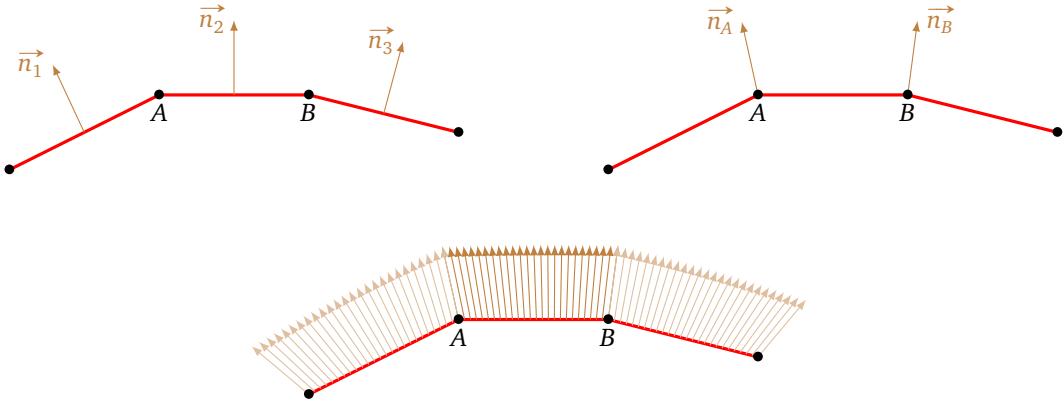
$$\vec{n}_P = (1 - t)\vec{n}_A + t\vec{n}_B$$



Il est abusif de parler de « vecteurs normaux » pour les vecteurs \vec{n}_P car ils ne sont en général pas orthogonaux à \vec{AB} . Cependant ils vont remplacer le vecteur normal lors des calculs d'éclairage et permettent de simuler une courbure entre A et B . En fait ce sont les vrais vecteurs normaux d'une arête courbe qui sont ici ramenés sur une arête rectiligne. On renvoie de nouveau au chapitre « Texture ».

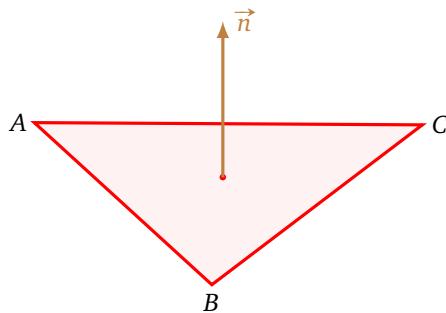


Plusieurs arêtes. Comment simuler une courbe lisse à partir de plusieurs arêtes ? (a) On détermine un (vrai) vecteur normal de chaque arête. (b) En chaque sommet on définit un vecteur comme la moyenne de deux vecteurs normaux des arêtes adjacentes. (c) Par interpolation linéaire on calcule un vecteur en chaque point de l'arête.



2.2. Normales à une face triangulaire

Une seule normale. Considérons un triangle ABC de l'espace. Sa normale (la vraie) est le vecteur \vec{n} donné par $\vec{AB} \wedge \vec{AC}$ (que l'on peut rendre unitaire si on préfère).

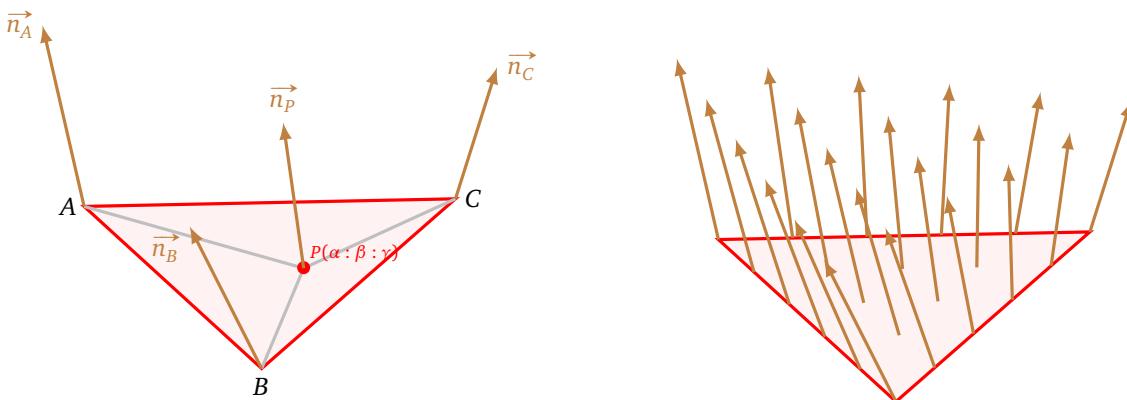


Une normale à chaque extrémité. Si on impose des vecteurs \vec{n}_A , \vec{n}_B et \vec{n}_C aux sommets alors les coordonnées barycentriques remplacent l'interpolation linéaire et permettent de définir un vecteur en n'importe quel point du triangle. On renvoie au chapitre « Texture » pour la définition de ces coordonnées. Pour un point du triangle, notons $(\alpha : \beta : \gamma)$ les coordonnées barycentriques d'un point P du triangle :

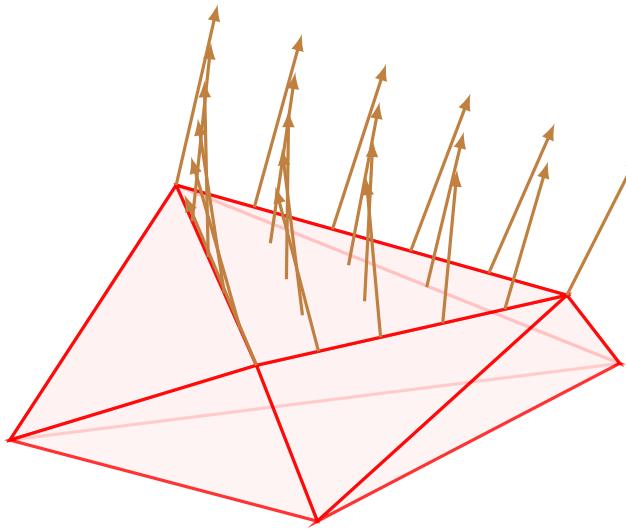
$$P = \alpha A + \beta B + \gamma C.$$

On définit alors le vecteur \vec{n}_P par ces mêmes coefficients :

$$\vec{n}_P = \alpha \vec{n}_A + \beta \vec{n}_B + \gamma \vec{n}_C.$$



Plusieurs triangles. Pour simuler une surface lisse à partir de plusieurs faces (triangulaires) planes. (a) On détermine un (vrai) vecteur normal de chaque face. (b) En chaque sommet on définit un vecteur comme la moyenne de vecteurs normaux des faces adjacentes. (c) Enfin on calcule un vecteur en chaque point de l'arête à l'aide des coordonnées barycentriques.



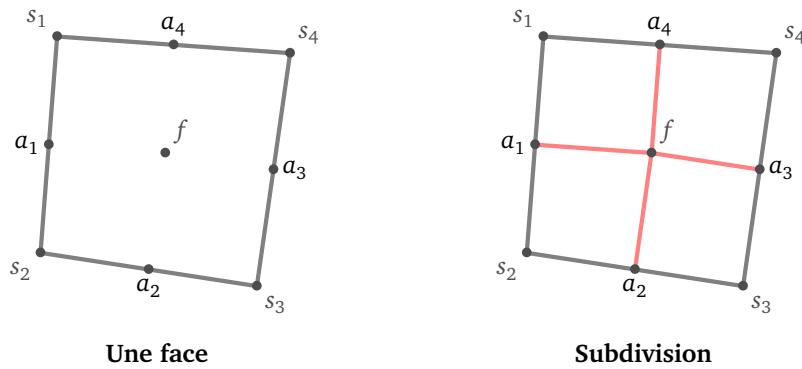
3. Subdivision

Une subdivision c'est partir d'un maillage simple pour obtenir un maillage avec plus de sommets, qui est donc censé mieux représenter la surface voulue. Nous allons décrire l'algorithme de Catmull-Clark qui est une méthode de subdivision par approximation : bien sûr à la fin nous obtiendrons de nouveaux sommets, mais les sommets originaux ne sont pas conservés à leur place d'origine.

3.1. Subdivision combinatoire

Commençons par décrire l'opération d'un point de vue combinatoire.

- Pour chaque face on considère son centre f , obtenu comme isobarycentre (c'est-à-dire la moyenne) des sommets s_1, s_2, \dots, s_n définissant cette face : $f = \frac{s_1+s_2+\dots+s_n}{n}$.
- Pour chaque arête de cette face on calcule son milieu a_j : si l'arête a pour extrémité s_{i_1} et s_{i_2} alors $a_j = \frac{s_{i_1}+s_{i_2}}{2}$.
- On remplace la face par n quadrilatères obtenus en ajoutant des arêtes reliant le centre f aux milieux a_j des arêtes.



Noter que même si la face de départ n'est pas un quadrilatère, les faces obtenues après subdivision sont toutes des quadrilatères. Dans la suite de la construction nous allons déplacer les sommets et donc, en général, les quatre sommets des quadrilatères ne seront pas situés dans un même plan.

3.2. Déplacement des sommets

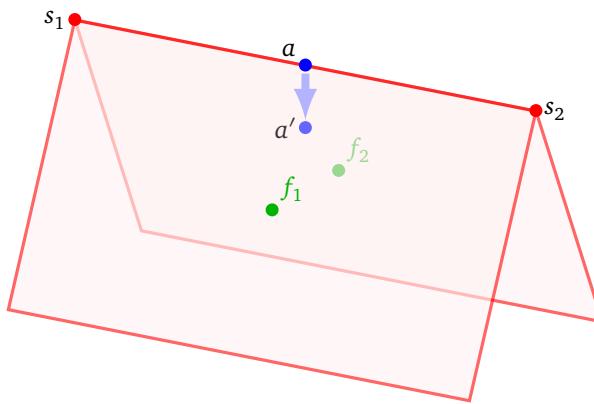
Nous allons maintenant déplacer les sommets. D'une part nous allons bouger les milieux des arêtes a_j que l'on vient de créer, mais en plus on va déplacer les sommets originaux s_i . Ces deux opérations sont indépendantes l'une de l'autre.

Déplacement des milieux des arêtes.

Considérons une arête d'extrémité les sommets s_1 et s_2 , notons a son milieu et f_1 et f_2 les centres des deux faces adjacentes. On remplace le sommet a par :

$$a' = \frac{s_1 + s_2 + f_1 + f_2}{4}$$

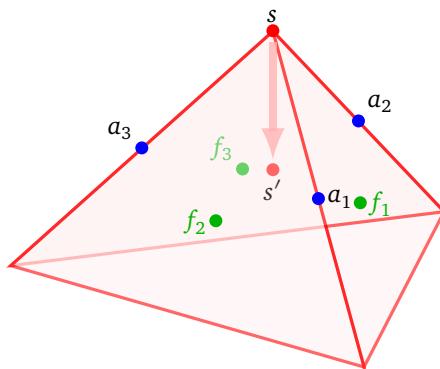
Ainsi a' (qui servira de sommet dans la subdivision en lieu et place de a) est l'isobarycentre de s_1 , s_2 , f_1 et f_2 . Comme $a = \frac{s_1 + s_2}{2}$, on peut aussi dire que a' est la moyenne entre : l'ancien a et la moyenne des centres des faces adjacentes.



Déplacement des sommets originaux.

Cas de 3 faces. Considérons un sommet s adjacent à exactement 3 faces de centres f_1 , f_2 , f_3 . Ce sommet est alors adjacent à 3 arêtes dont on note les milieux a_1 , a_2 , a_3 (ce sont les vrais milieux a_j et pas les points déplacés a'_j). On calcule la moyenne F des centres des faces, la moyenne A des milieux des arêtes et on effectue une moyenne pondérée pour le nouveau sommet s' (qui remplace le sommet original s) :

$$F = \frac{f_1 + f_2 + f_3}{3} \quad A = \frac{a_1 + a_2 + a_3}{3} \quad s' = \frac{F + 2A}{3}$$

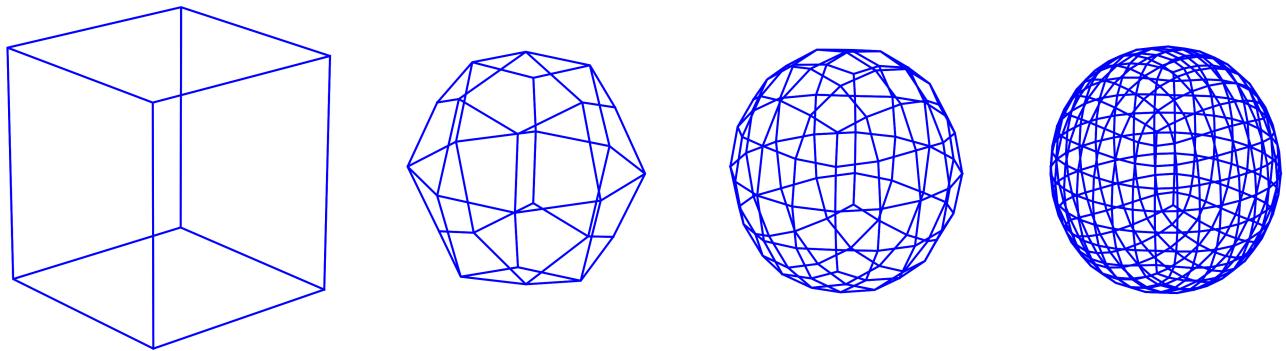


Cas de n faces. Dans le cas de n faces arrivant en un sommet s on remplace ce sommet s par un sommet s' donné par les formules suivantes :

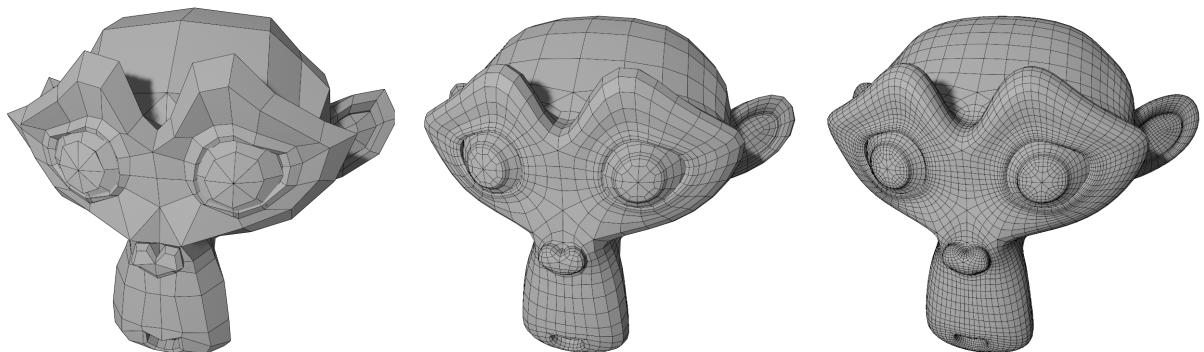
$$F = \frac{f_1 + \dots + f_n}{n} \quad A = \frac{a_1 + \dots + a_n}{n} \quad s' = \frac{F + 2A + (n-3)s}{n}$$

En termes de barycentre, s' est le barycentre de sommets f_1, \dots, f_n affectés chacun d'un poids 1 et de sommets a_1, \dots, a_n affectés chacun d'un poids 2 et du sommet original s affecté d'un poids $n - 3$.

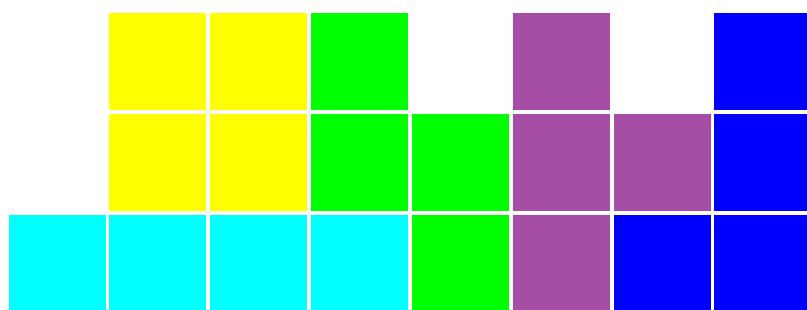
Ci-dessous de gauche à droite : (a) l'objet original est un cube (avec 6 faces), (b) on applique l'algorithme de Catmull-Clark une première fois, chaque face du cube est remplacée par 4 quadrilatères, on obtient un objet à 24 faces, (c) on itère le processus, (d) on itère encore. Chaque itération fournit un objet plus lisse.



De gauche à droite : (a) l'objet original, (b) une application de l'algorithme de subdivision, (c) une seconde application.



TROISIÈME PARTIE



APPLICATIONS

Mouvement

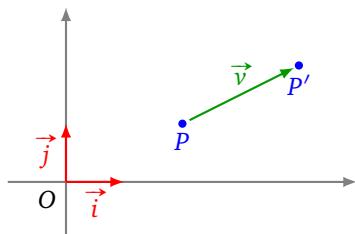
Comment se déplacer dans le plan, dans l'espace, dans un labyrinthe, sur un terrain ?

1. Position, vitesse, accélération

1.1. Position

Déplacement via les coordonnées cartésiennes. Dans le plan on repère la position d'un objet ponctuel par deux coordonnées $P = (x, y)$. Pour se déplacer, on peut indiquer le vecteur déplacement souhaité $\vec{v} = (v_x, v_y)$. Le nouveau point est :

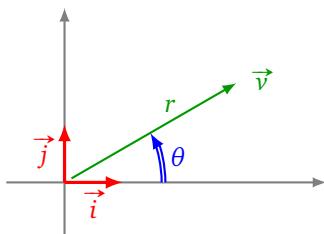
$$P' = P + \vec{v} = (x + v_x, y + v_y).$$



Autrement dit, pour aller du point $P_1 = (x_1, y_1)$ au point $P_2 = (x_2, y_2)$, le vecteur correspondant est :

$$\vec{v} = P_2 - P_1 = (x_2 - x_1, y_2 - y_1).$$

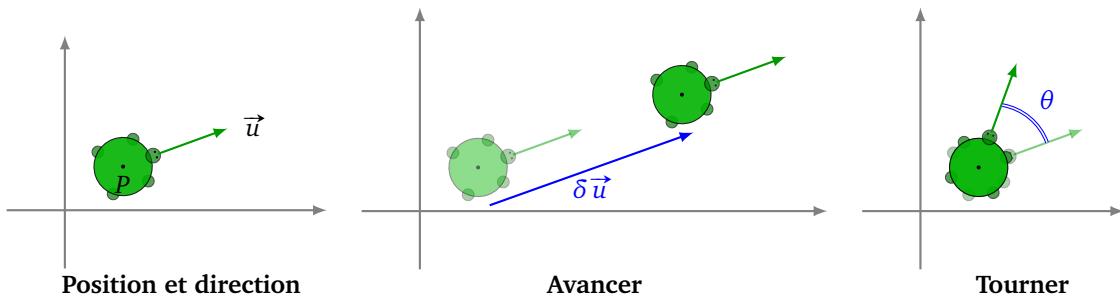
Déplacement via les coordonnées polaires. On pourrait aussi repérer \vec{v} par ses coordonnées polaires $[r : \theta]$ c'est-à-dire $\vec{v} = (r \cos \theta, r \sin \theta)$.



Déplacement en mode « tortue ». On repère une tortue en déplacement (comme dans le logiciel Scratch) par sa position actuelle P et sa direction \vec{u} .

- L'instruction *avancer* correspond à déplacer la tortue au point $P' = P + \delta \vec{u}$ où δ correspond à un nombre de pas par exemple.
- L'instruction *tourner* correspond à changer le vecteur de direction \vec{u} , en lui appliquant une rotation d'angle θ . Si \vec{u} est un vecteur unitaire (c'est-à-dire $\|\vec{u}\| = 1$) alors le nouveau vecteur de direction est :

$$\vec{u}' = \vec{u} + (\cos \theta, \sin \theta).$$



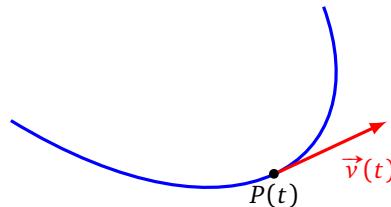
Dans l'espace. Nous avons besoin de trois coordonnées $P = (x, y, z)$ et $\vec{v} = (v_x, v_y, v_z)$ et alors $P' = P + \vec{v} = (x + v_x, y + v_y, z + v_z)$. Pour comprendre comment se déplacer en mode « tortue » dans l'espace, on renvoie au chapitre « Fractales ».

1.2. Vitesse

Considérons un point $P(t)$ en mouvement en fonction d'un paramètre de temps $t \in \mathbb{R}$. La **vitesse** $\vec{v}(t)$ est définie comme la dérivée de la position par rapport au temps :

$$\vec{v}(t) = \frac{dP(t)}{dt}.$$

Le vecteur vitesse $\vec{v}(t)$ est tangent à la trajectoire en $P(t)$ et plus la particule se déplace vite, plus sa longueur est grande.



Si la position $P(t)$ est donnée en coordonnées par $(x(t), y(t))$, calculer le vecteur dérivé revient à dériver chacune des fonctions $t \mapsto x(t)$ et $t \mapsto y(t)$:

$$\vec{v}(t) = (x'(t), y'(t)).$$

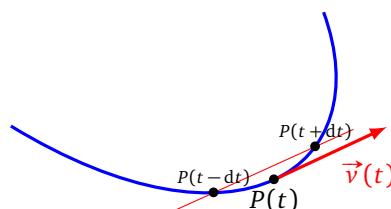
Pour estimer cette vitesse instantanée on peut calculer une vitesse moyenne sur un intervalle de temps dt très court :

$$\vec{v}(t) \simeq \frac{P(t + dt) - P(t)}{dt}.$$

Cette formule correspond à la définition de la dérivée comme une limite.

Remarque : une meilleure approximation peut être obtenue par la formule :

$$\vec{v}(t) \simeq \frac{P(t + dt) - P(t - dt)}{2dt}.$$



1.3. Accélération

L'**accélération** est la dérivée de la vitesse :

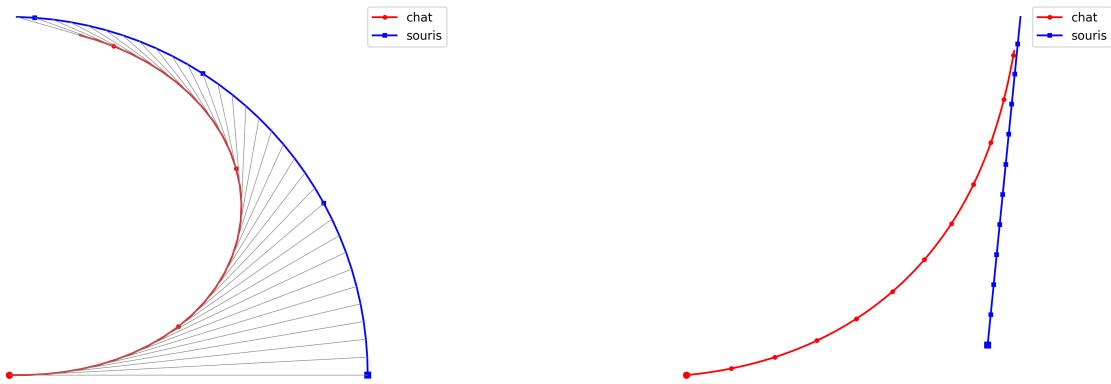
$$\vec{a}(t) = \frac{d\vec{v}(t)}{dt}.$$

En coordonnées : $\vec{a}(t) = (x''(t), y''(t))$.

Par exemple, un objet soumis à aucune force aura une accélération égale au vecteur nul et se déplace suivant un mouvement rectiligne uniforme : son vecteur vitesse est constant $\vec{v}(t) = \vec{v}_0$ et sa position à l'instant t sera $P(t) = P(0) + t\vec{v}_0$.

1.4. Courbe de poursuite

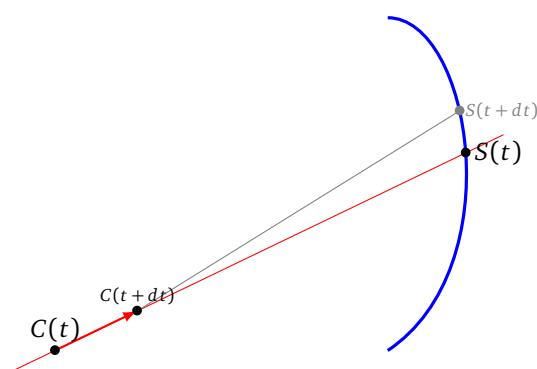
Tracer des courbes de poursuite est un petit exercice de programmation amusant. Un chat C court après une souris S . La souris a une trajectoire $S(t)$ et à tout instant le chat se dirige vers la souris. Tracer la trajectoire $C(t)$ du chat.



Les données et les étapes sont les suivantes :

- se donner une trajectoire $S(t)$ pour la souris,
- se donner une position initiale $C(0)$ du chat,
- se donner un réel v pour la norme de la vitesse du chat,
- définir un intervalle élémentaire de temps dt (par exemple $dt = 0.1$),
- faire une boucle correspondant au déroulement du temps :
 - une fois le chat en position $C(t)$, calculer le vecteur $\vec{C(t)S(t)}$,
 - faire $C(t + dt) = C(t) + v \frac{\vec{C(t)S(t)}}{\|\vec{C(t)S(t)}\|}$.

On obtient ainsi une liste de points C_i correspondant à la trajectoire du chat.



2. Mouvement circulaire

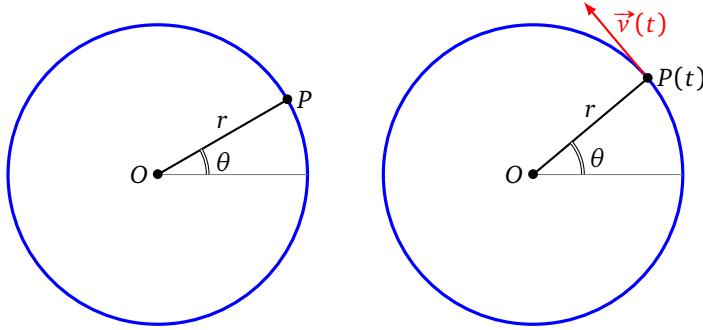
2.1. Position, vitesse, accélération

Position. On considère un point P situé sur un cercle de rayon r , centré à l'origine $O = (0, 0)$. La position de ce point est alors déterminée par l'angle θ formé par les vecteurs \vec{i} et \vec{OP} .

$$P = (x, y) \quad \text{avec} \quad x = r \cos \theta \quad \text{et} \quad y = r \sin \theta.$$

Considérons maintenant un point P en mouvement, tout en restant sur le cercle, comme par exemple si P était relié à une barre rigide tournant autour de O . L'angle $\theta(t)$ dépend du temps t et la position P est :

$$P = (x(t), y(t)) \quad x(t) = r \cos \theta(t) \quad y(t) = r \sin \theta(t).$$



Vitesse. La vitesse est alors : $\vec{v}(t) = \frac{dP(t)}{dt} = (x'(t), y'(t))$ où

$$x'(t) = -r\theta'(t)\sin(\theta(t)) \quad y'(t) = r\theta'(t)\cos(\theta(t)).$$

Comparons le vecteur vitesse et le vecteur position :

$$\overrightarrow{OP(t)} = \begin{pmatrix} r \cos \theta(t) \\ r \sin \theta(t) \end{pmatrix} \quad \vec{v}(t) = \begin{pmatrix} -r\theta'(t)\sin \theta(t) \\ r\theta'(t)\cos \theta(t) \end{pmatrix}$$

On voit facilement que ces deux vecteurs sont orthogonaux car leur produit scalaire $\overrightarrow{OP(t)} \cdot \vec{v}(t)$ est nul (quel que soit t). Cela signifie que la vitesse est perpendiculaire au rayon en P , autrement dit le vecteur vitesse est tangent au cercle.

Accélération. On calculerait l'accélération par la formule $\vec{a}(t) = (x''(t), y''(t))$. On décompose souvent le vecteur en une composante radiale et une composante tangentielle $\vec{a}(t) = \vec{a}_{\text{rad}}(t) + \vec{a}_{\text{tan}}(t)$.

2.2. Mouvement circulaire uniforme

Définition. Lorsque la vitesse angulaire est constante, c'est-à-dire $\theta(t) = \omega t + \theta_0$, le mouvement est dit **circulaire uniforme**. En effet $\theta'(t) = \omega$ (la vitesse angulaire) est constante et donc l'accélération angulaire est nulle : $\theta''(t) = 0$.

Vecteur vitesse. Calculons le vecteur vitesse $\vec{v}(t)$ dans ce cas. Pour alléger l'écriture, on fixe $\theta_0 = 0$.

$$\vec{v}(t) = (x'(t), y'(t)) = (-r\omega \sin(\theta t), r\omega \cos(\theta t)).$$

La norme vaut $\|\vec{v}(t)\| = r|\omega|$.

Accélération.

$$\vec{a}(t) = (x''(t), y''(t)) = (-r\omega^2 \cos(\theta t), -r\omega^2 \sin(\theta t)) = -\omega^2 \overrightarrow{OP(t)}.$$

Ainsi l'accélération dans un mouvement circulaire uniforme est uniquement radiale (l'accélération tangentielle est nulle). Cela correspond à une force centripète (une force centrifuge étant ressentie au point en mouvement).

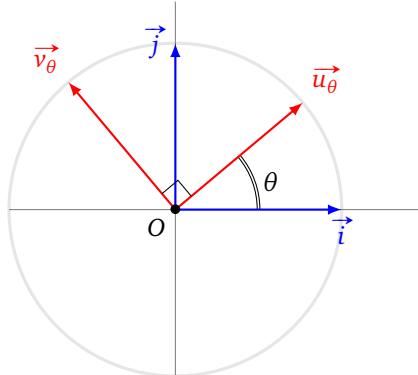
Période. La **période** $T = \frac{2\pi}{\omega}$ est le temps (en secondes) afin qu'un point revienne à sa position de départ après un tour complet. La **fréquence** $f = \frac{1}{T} = \frac{\omega}{2\pi}$ est le nombre de tours par seconde.

2.3. Moment d'une force

Action d'une force. On considère une barre $[OP]$ rigide de longueur r qui peut tourner autour de O . Que se passe-t-il lorsqu'on applique une force \vec{F} en P ? La barre va se mettre à tourner, sauf dans le cas où la force est parallèle à \overrightarrow{OP} . On mesure l'action de cette force grâce au *moment*.

Repère tournant. Le repère tournant d'angle θ est le repère orthonormal $(O, \vec{u}_\theta, \vec{v}_\theta)$ défini par :

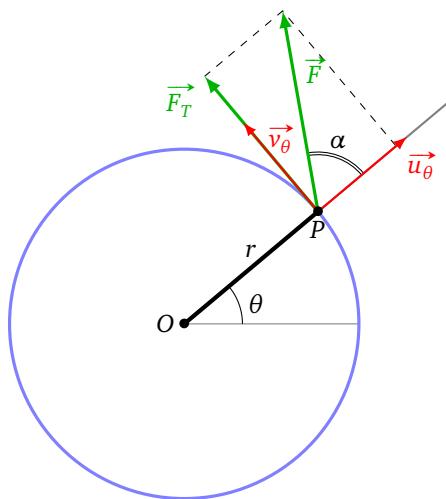
$$\vec{u}_\theta = (\cos \theta, \sin \theta) \quad \vec{v}_\theta = (-\sin \theta, \cos \theta).$$



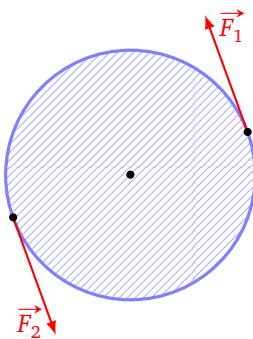
Moment (scalaire) d'une force. Le *moment* d'une force \vec{F} par rapport au point P est défini par :

$$m = r \vec{F} \cdot \vec{v}_\theta = r F \sin \alpha$$

où θ est l'angle de \overrightarrow{OP} avec l'horizontale et α est l'angle entre \vec{u}_θ et \vec{F} . Géométriquement la seule force qui actionne la barre est la force tangentielle \vec{F}_T , qui est la projection de la force \vec{F} sur la tangente au cercle en P . Le moment est la longueur de cette force tangentielle : $|m| = \|\vec{F}_T\|$.



Couple. Si on considère maintenant un disque rigide de rayon r , on peut lui appliquer deux forces opposées \vec{F}_1 et \vec{F}_2 (voir le dessin). La résultante des forces est nulle, cependant le disque tourne. Cela s'explique car les deux moments ont le même signe (en fait ils sont égaux) $m_1 = m_2 = rF$. C'est la notion de *couple*.

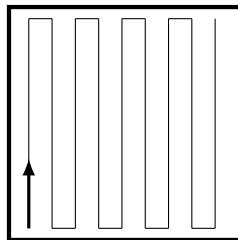


3. Labyrinthe

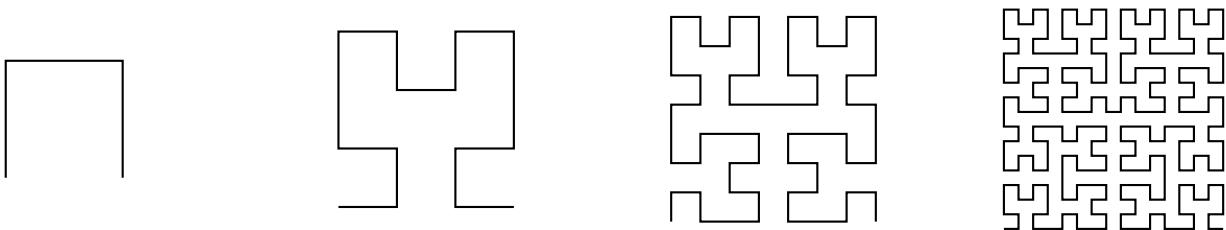
Dans cette section nous nous plaçons du point de vue du joueur/personnage qui doit trouver un objet ou la sortie en ayant une vue limitée de son environnement.

3.1. Balayage

Tout d'abord notre joueur doit trouver un trésor dans une pièce carrée sans obstacle. S'il n'a pas d'indice, il peut entamer une recherche systématique par balayage, en s'approchant d'autant près de tout point que nécessaire.

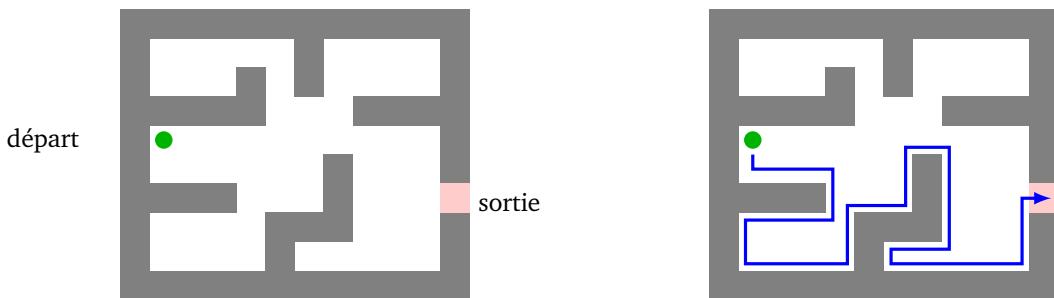


Il existe d'autres chemins, beaucoup plus tortueux, comme la courbe de Hilbert qui a une structure fractale et s'obtient par une formule de substitution. On renvoie au chapitre « Fractales » pour plus de détails et aussi sa version 3D. Ci-dessous les itérations d'ordre 1,2,3,4 vers la courbe d'Hilbert :

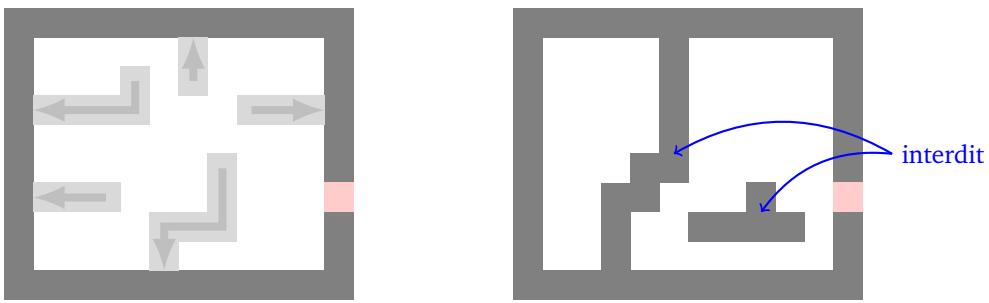


3.2. Labyrinthe

Considérons maintenant notre personnage perdu dans un labyrinthe. Une façon d'en sortir est de se déplacer au hasard. La probabilité de sortir vaut 1, mais cela peut être très long. Dans le cas d'un labyrinthe aux murs *simplement connexes* la technique de *la main droite* permet toujours de trouver la sortie. Il s'agit d'avancer en longeant la paroi de sorte que la main droite touche toujours le mur.



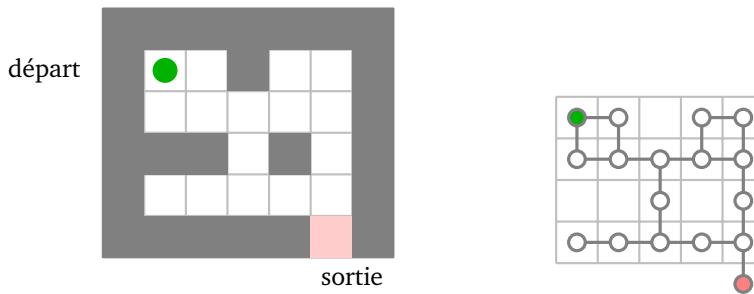
Être *simplement connexe* signifie que l'on peut contracter les murs intérieurs sur le bord (la sortie étant dans un bord). En particulier il n'y a pas de murs isolés au milieu du labyrinthe. Lorsqu'on contracte les murs intérieurs on se convainc facilement que cette méthode fonctionne.



3.3. Graphe

Transformons un labyrinthe quelconque en un graphe :

- chaque case du labyrinthe est un sommet,
- si deux cases sont adjacentes, les sommets correspondants sont reliés entre eux par une arête.



On marque un sommet comme départ, un autre comme arrivée. Il s'agit de trouver un chemin dans le graphe reliant le départ à l'arrivée. (On renvoie au chapitre « Triangulation » pour le vocabulaire sur les graphes.) Sortir du labyrinthe est donc maintenant un problème de parcours de graphe. Il existe de nombreux algorithmes pour cela. Noter que dans le cas d'un labyrinthe aux murs simplement connexes, et si les couloirs ont une largeur d'une seule case, alors le graphe est en fait un arbre. (Voir le chapitre « Minimax » pour la définition d'arbre et leur parcours en largeur ou en profondeur.)

Revenons au cas général : nous allons voir une méthode pour relier deux sommets qui est en fait une version simplifiée de l'algorithme de Dijkstra.

Le procédé se décompose en deux étapes :

- Trouver la distance entre n'importe quel sommet et le sommet de départ.
- Puis partir de la sortie et rebrousser chemin vers le départ, en diminuant la distance calculée à chaque pas.

L'algorithme utilise une file qui contient la liste des sommets en cours de traitement. Une file (*fifo*) est analogue à une pile (*queue/filo*) sauf que l'on retire le premier élément mis dans la file (et pas le dernier).

Algorithme (Distances au départ).

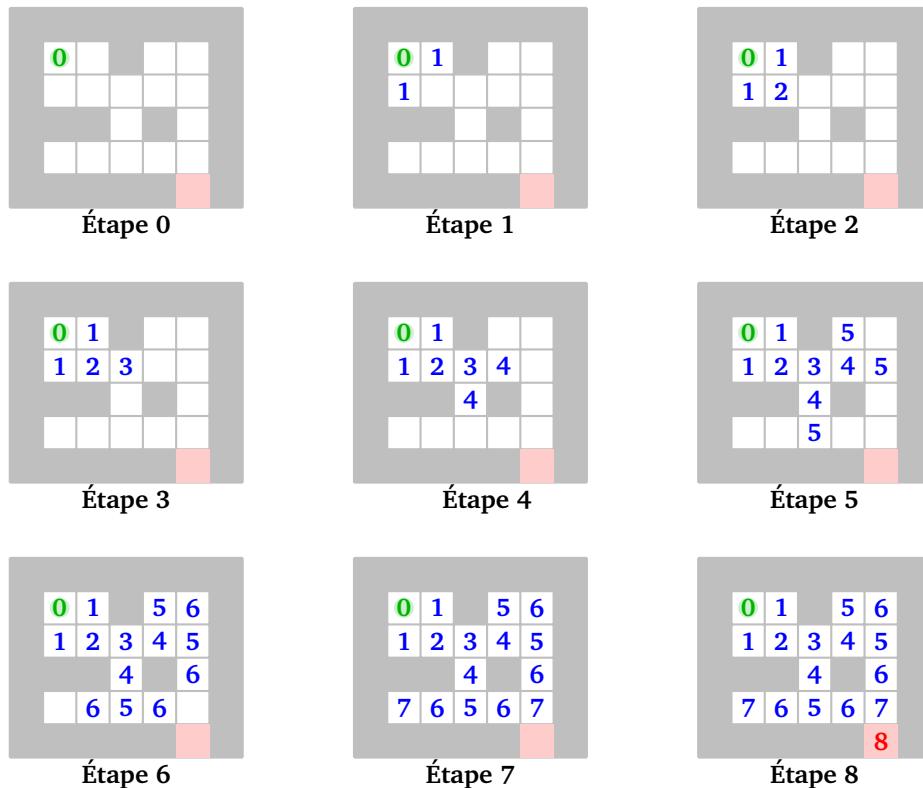
Entrée : un graphe (connexe) avec un sommet « départ ».

Sortie : pour chaque sommet s , la longueur $d(s)$ du plus petit chemin le reliant au départ.

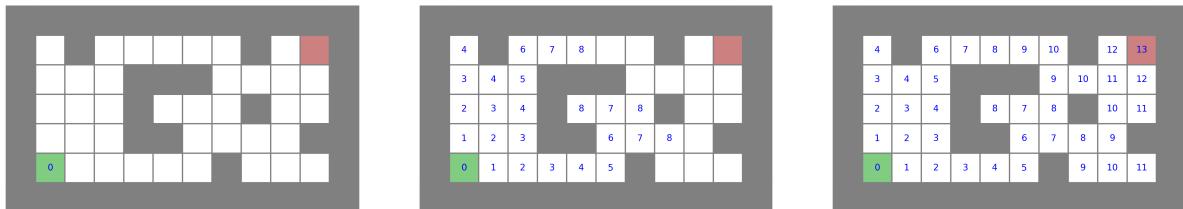
- Mettre le sommet s de « départ » dans la file et pour ce sommet poser $d(s) = 0$.
- Tant que la file n'est pas vide :
 - prendre s le premier sommet de la file (et le retirer de la file),
 - pour chaque voisin s' de s qui n'a pas encore été visité :
 - poser $d(s') = d(s) + 1$
 - ajouter s' à la fin de la file.

Voici comment se déroule l'algorithme : les voisins s de la case de départ se voient attribuer la valeur 1 et sont placés dans la file ; on prend ensuite un de ces voisins s , les voisins s' de s se voient attribuer la valeur

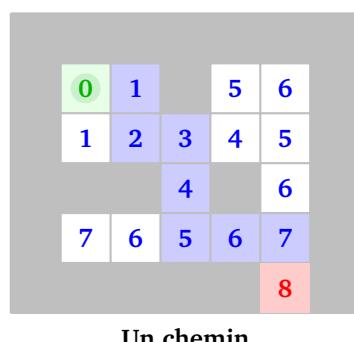
2 et sont placés dans la file (après ceux de valeur 1) ; on repart d'un des voisins s de la case de départ et on fait le même traitement jusqu'à ce que les voisins de valeur 1 soient tous traités, ensuite la file commence par les sommets de valeur 2, etc.



Voici un exemple plus compliqué :



Pour obtenir un chemin et la distance reliant le départ et l'arrivée il suffit de partir de la fin et de chercher à chaque étape un voisin strictement plus proche du départ.



Un chemin

Algorithme (Chemin reliant le départ et l'arrivée).

Entrée : un graphe, un sommet « départ », un sommet « arrivée » et pour chaque sommet du graphe sa distance au sommet « départ ».

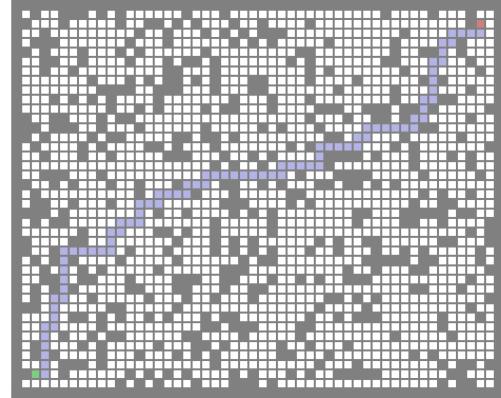
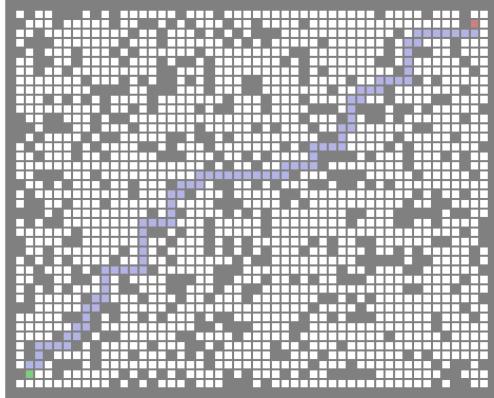
Sortie : un chemin reliant le départ à l'arrivée.

- Se placer au sommet s de l'arrivée, initialiser une liste **chemin** à s .
- Tant que s n'est pas le sommet de départ :
 - parmi les voisins de s , choisir s' tel que $d(s') < d(s)$,
 - ajouter s' à **chemin**,
 - faire $s \leftarrow s'$.

Lors du choix de s' parmi les voisins de s vérifiant $d(s') < d(s)$ il y a (par construction) au moins une possibilité, s'il y en a plusieurs, on peut choisir n'importe laquelle (au hasard par exemple).

4	6	7	8	9	10	12	13
3	4	5		9	10	11	12
2	3	4	8	7	8	10	11
1	2	3		6	7	8	9
0	1	2	3	4	5	9	10

4	6	7	8	9	10	12	13
3	4	5		9	10	11	12
2	3	4	8	7	8	10	11
1	2	3		6	7	8	9
0	1	2	3	4	5	9	10

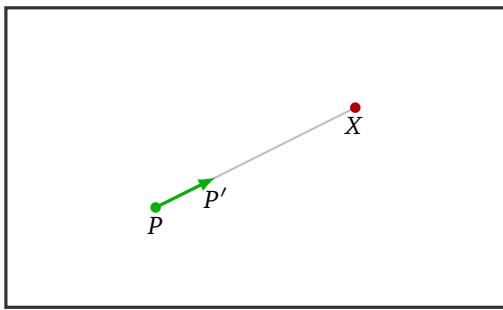


4. Terrain

4.1. Modélisation

Un personnage à la position P doit atteindre l'objectif X . Si le terrain est dégagé alors il se déplace en ligne droite en direction de l'objectif. S'il avance d'un pas de longueur δ alors sa nouvelle position est :

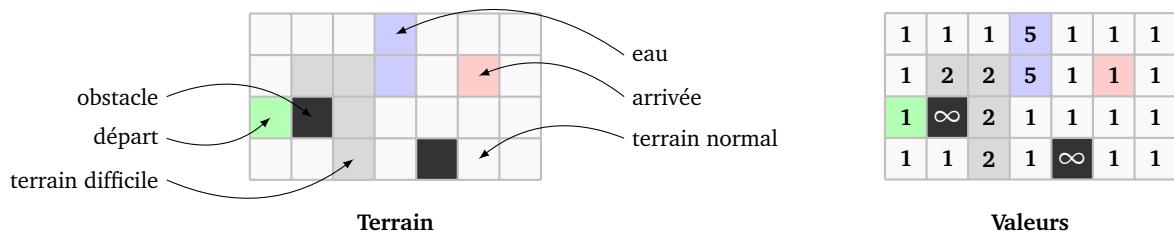
$$P' = P + \delta \frac{\overrightarrow{PX}}{\|\overrightarrow{PX}\|}.$$



Mais comment se déplacer s'il y a des obstacles sur son trajet? Et si le terrain n'est pas uniforme, par exemple avec des zones où l'avancement est plus difficile : faut-il mieux foncer tout droit ou contourner les difficultés?

Pour répondre à ces questions, nous allons modéliser un terrain par une grille rectangulaire. Chaque case peut avoir un type de terrain différent. À chaque type de terrain on attribue une valeur. Plus la valeur est élevée, plus le terrain est difficile et donc la progression plus lente. Voici les types de terrain qu'on va rencontrer :

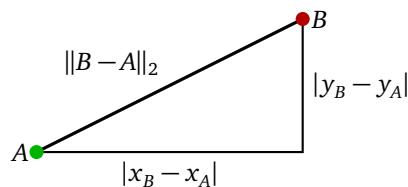
- terrain facile : valeur 1,
- terrain difficile : valeur 2 (on s'y déplace deux fois moins vite),
- eau/lac : valeur 5 (on peut nager, mais c'est lent),
- obstacle infranchissable : valeur infinie (dans la pratique il suffit de mettre une valeur élevée).



Dans cette modélisation on autorise les déplacements en diagonale.

4.2. Distances

Il y a plusieurs façons de mesurer la distance entre deux points. Ce choix est important.



Distance de Manhattan. La **norme 1** est définie par $\|(x, y)\|_1 = |x| + |y|$. La distance associée s'appelle la distance de Manhattan : $d_1(A, B) = \|B - A\|_1 = |x_B - x_A| + |y_B - y_A|$, car elle correspond à la distance qu'il faut parcourir lorsqu'on se déplace sur une grille.

Pour une case, ses voisins immédiats sont donc les 4 cases situées immédiatement à gauche/droite/haut/bas, les centres des cases placées en diagonale sont à une distance 2 du centre de la case centrale.

2	1	2
1	0	1
2	1	2

Distance 1
Distance de Manhattan

$\sqrt{2}$	1	$\sqrt{2}$
1	0	1
$\sqrt{2}$	1	$\sqrt{2}$

Distance 2
Distance euclidienne

1	1	1
1	0	1
1	1	1

Distance infinie
Distance de Tchebychev

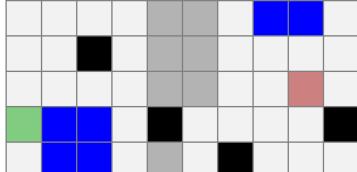
Distance euclidienne. La **norme 2** est définie par $\|(x, y)\|_2 = \sqrt{x^2 + y^2}$. La distance associée s'appelle la distance euclidienne : $d_2(A, B) = \|B - A\|_2 = \sqrt{(x_B - x_A)^2 + (y_B - y_A)^2}$, c'est la distance usuelle « à vol d'oiseau ». Cette fois les centres des cases en diagonale sont à une distance $\sqrt{2}$ du centre de la case centrale.

Distance de Tchebychev. La **norme infinie** est définie par $\|(x, y)\|_\infty = \max(|x|, |y|)$. La distance associée s'appelle la distance de Tchebychev : $d_\infty(A, B) = \|B - A\|_\infty = \max(|x_B - x_A|, |y_B - y_A|)$. Avec cette distance, les cases en diagonale sont des voisins immédiats de la case centrale.

4.3. Heatmap

On fixe l'une des distances précédentes. Pour se diriger on commence par calculer une *heatmap*, c'est-à-dire une carte de chaleur. Plus une case est rouge plus elle est proche de l'objectif, les teintes bleues correspondent aux cases plus éloignées.

Ci-dessous, de gauche à droite : (a) le terrain, (b) la distance à la case d'arrivée (selon la distance de Manhattan), (c) les couleurs de la *heatmap*.



12.0	11.0	10.0	9.0	8.0	6.0	4.0	7.0	6.0	3.0
11.0	10.0	—	8.0	7.0	5.0	3.0	2.0	1.0	2.0
10.0	9.0	8.0	7.0	6.0	4.0	2.0	1.0	0.0	1.0
11.0	10.0	13.0	8.0	—	4.0	3.0	2.0	1.0	—
12.0	17.0	13.0	8.0	7.0	5.0	—	3.0	2.0	3.0

12.0	11.0	10.0	9.0	8.0	6.0	4.0	7.0	6.0	3.0
11.0	10.0	—	8.0	7.0	5.0	3.0	2.0	1.0	2.0
10.0	9.0	8.0	7.0	6.0	4.0	2.0	1.0	0.0	1.0
11.0	14.0	13.0	8.0	—	4.0	3.0	2.0	1.0	—
12.0	17.0	13.0	8.0	7.0	5.0	—	3.0	2.0	3.0

Concrètement la *heatmap* est une fonction h qui à chaque case (i, j) lui associe sa distance à l'objectif. Si on passe par une case de valeur 1, la distance augmente de 1, si on passe sur un terrain plus difficile, par exemple une case d'eau, la distance augmente de 5. On tient compte en plus de la distance entre deux cases voisines (qui n'est pas toujours 1 pour les cases en diagonales).

L'algorithme du calcul de la *heatmap* est similaire au calcul de la distance utilisé pour les labyrinthes mais il y a un subtilité supplémentaire car les distances associées à chaque case dépendent du terrain.

Ci-dessous on commence par compléter les distances case par case. On commence par la case d'arrivée (en rouge, figure (b)) qui a pour valeur de la *heatmap* 0. On atteint assez vite la case de départ (en vert, figure (c)), mais lorsqu'on continue de compléter les distances on s'aperçoit qu'il existe un chemin plus rapide même si on parcourt plus de cases (figures (d) et (e)) et cela oblige à mettre à jour la distance entre la case de départ et celle d'arrivée.

1	5	5	1
1	5	1	1
1	1	1	1

(a) Terrain

	11	6	1
6	1	0	
3	2	1	

(b) Début du calcul de la heatmap

	11	6	1
7	6	1	0
3	2	1	

(c) Première valeur à la case verte

	11	6	1
7	6	1	0
4	3	2	1

(d) On continue

	11	6	1
5	6	1	0
4	3	2	1

(e) Mise à jour de la valeur à la case verte

6	11	6	1
5	6	1	0
4	3	2	1

(f) La heatmap complétée

Comme auparavant on identifie la grille à un graphe : une case correspond à un sommet, les sommets adjacents correspondent aux 8 cases voisines (y compris les cases en diagonale).

Algorithme (Calcul de la heatmap).

Entrée : un graphe (connexe) avec un sommet « arrivée », la valeur $t(s)$ du terrain associée à chaque sommet et le choix d'une distance d .

Sortie : pour chaque sommet s , la longueur $h(s)$ du plus petit chemin le reliant au départ.

- Pour tous les sommets, initialiser $h(s)$ à $+\infty$.
- Mettre le sommet s d'« arrivée » dans la file et pour ce sommet poser $h(s) = 0$.
- Tant que la file n'est pas vide :
 - prendre s le premier sommet de la file (et le retirer de la file),
 - pour chaque voisin s' de s (même ceux déjà visités) :
 - calculer $H = h(s) + d(s,s') \times t(s')$
 - si $H < h(s')$, alors :
 - poser $h(s') = H$,
 - ajouter s' à la fin de la file.

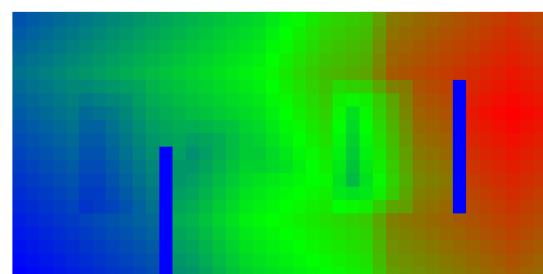
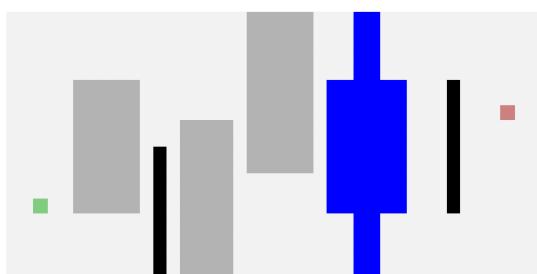
Ci-dessous, de gauche à droite, les valeurs de la heatmap pour les normes 1, 2 et infinie.

12.0	11.0	10.0	9.0	8.0	6.0	4.0	7.0	6.0	3.0
11.0	10.0	9.0	8.0	7.0	5.0	3.0	2.0	1.0	2.0
10.0	9.0	8.0	7.0	6.0	4.0	2.0	1.0	0.0	1.0
11.0	10.0	13.0	8.0	7.0	4.0	3.0	2.0	1.0	2.0
12.0	11.0	13.0	8.0	7.0	5.0	3.0	2.0	2.0	3.0

10.8	9.8	8.8	7.8	6.8	4.8	2.8	0.8	6.0	2.4
10.4	9.4	8.0	7.4	6.4	4.4	2.4	1.4	1.0	1.4
10.0	9.0	8.0	7.0	6.0	4.0	2.0	1.0	0.0	1.0
10.4	10.0	12.0	7.2	6.0	3.4	2.4	1.4	1.0	2.0
11.4	10.4	11.6	6.8	5.8	3.8	2.4	2.0	2.0	2.4

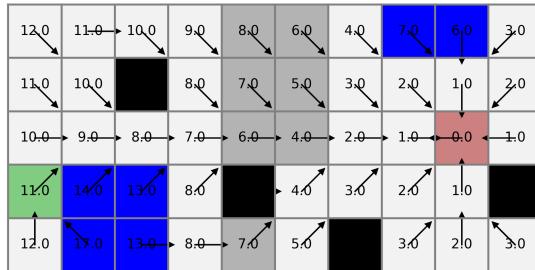
9.0	8.0	7.0	7.0	6.0	4.0	2.0	6.0	6.0	2.0
9.0	8.0	9.0	6.0	6.0	4.0	2.0	1.0	1.0	1.0
9.0	8.0	7.0	6.0	5.0	4.0	2.0	1.0	0.0	1.0
9.0	12.0	11.0	6.0	5.0	3.0	2.0	1.0	1.0	2.0
10.0	14.0	11.0	6.0	5.0	3.0	2.0	2.0	2.0	2.0

Ci-dessous un exemple plus compliqué : à gauche le terrain, à droite la coloration de la heatmap (pour la norme 1).



4.4. Chemin

Comme dans le cas du labyrinthe, pour se diriger vers l'arrivée, le plus simple est de chercher une case voisine avec la valeur de la *heatmap* la plus petite possible. Autrement dit à chaque case, une flèche indique la direction de la case voisine (y compris en diagonale) qui diminue le plus la distance vers l'arrivée. (S'il y a plusieurs cases qui réalisent ce minimum, on en prend une au hasard.)



4.5. Gradient

Avec la méthode du minimum, pour chaque case on obtient une direction parmi 8 possibles (correspondant aux 8 cases voisines), ce qui est assez limité. Comment obtenir plus de possibilités ?

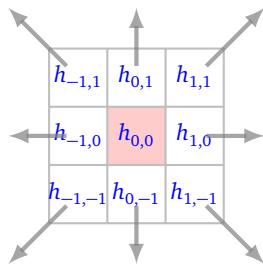
Le *gradient* d'une fonction h de deux variables (ou plus) indique la direction de la plus forte pente, c'est-à-dire la direction à suivre pour passer à une valeur de h la plus grande possible. C'est une version à deux dimensions de la dérivée. Noter qu'ici on veut minimiser h , donc on va en fait suivre la direction opposée au gradient.

Nous allons calculer le gradient de h en une case à l'aide d'une convolution vectorielle, c'est-à-dire qu'on va calculer un vecteur à l'aide de la grille 3×3 entourant la case. Depuis la case centrale, de coordonnées $(0, 0)$, on se déplace à une case (i, j) voisine selon un vecteur $\vec{v}_{ij} = (i, j)$ où $i, j \in \{-1, 0, 1\}$. Par exemple $(1, 0)$ désigne la case de droite et est associée au vecteur horizontal $(1, 0)$.

En chaque case nous avons aussi une valeur $h(i, j)$ donnée par la *heatmap*. Ce qui va intervenir c'est la différence entre la valeur h_{00} en la case centrale et la valeur h_{ij} en une case voisine.

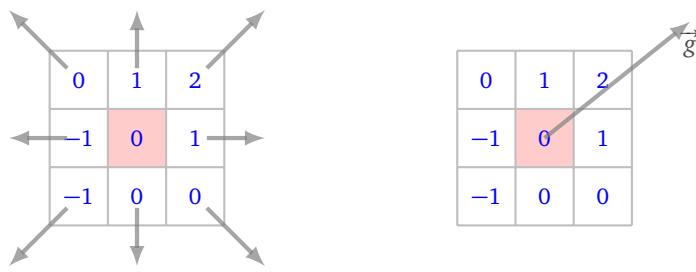
Le gradient en $(0, 0)$ se calcule selon la formule :

$$\vec{g} = \sum_{i,j} (h_{ij} - h_{00}) \vec{v}_{ij} = (h_{10} - h_{00}) \begin{pmatrix} 1 \\ 0 \end{pmatrix} + (h_{11} - h_{00}) \begin{pmatrix} 1 \\ 1 \end{pmatrix} + (h_{01} - h_{00}) \begin{pmatrix} 0 \\ 1 \end{pmatrix} + \dots$$



Exemple.

Partons de cette mini-*heatmap* (pour simplifier les calculs on a imposé $h_{00} = 0$).



Le vecteur gradient associé à la case centrale est :

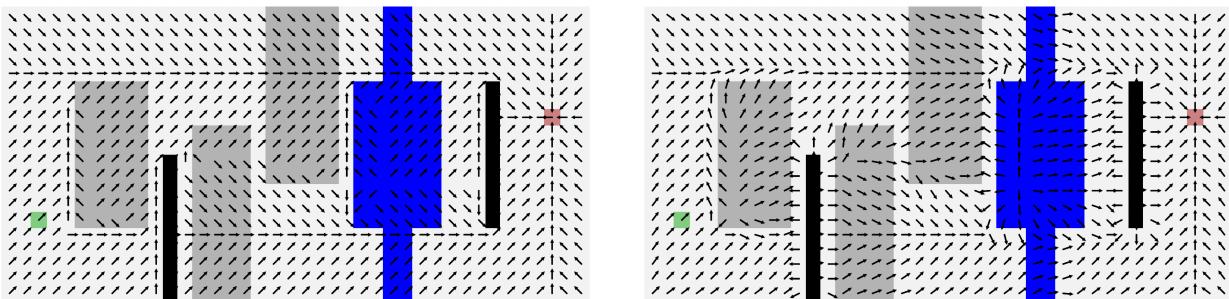
$$\begin{aligned}\vec{g} &= (1) \times \rightarrow + (2) \times \nearrow + (1) \times \uparrow + (0) \times \nwarrow + (-1) \times \leftarrow + (-1) \times \swarrow + (0) \times \downarrow + (0) \times \searrow \\ &= 1 \begin{pmatrix} 1 \\ 0 \end{pmatrix} + 2 \begin{pmatrix} 1 \\ 1 \end{pmatrix} + 1 \begin{pmatrix} 0 \\ 1 \end{pmatrix} - 1 \begin{pmatrix} -1 \\ 0 \end{pmatrix} - 1 \begin{pmatrix} -1 \\ -1 \end{pmatrix} \\ &= \begin{pmatrix} 5 \\ 4 \end{pmatrix}\end{aligned}$$

On se déplace ensuite vers la case indiquée par la direction \vec{g} . Noter que si $h_{ij} > h_{00}$ on obtient une contribution dans le sens \vec{v}_{ij} (on va donc se diriger vers des valeurs de h plus hautes), mais si $h_{ij} < h_{00}$ on obtient une contribution en sens inverse (on se dirige dans la direction opposée des valeurs plus basses). Quelques ajustements peuvent être nécessaires, car ici notre fonction présente des sauts importants au niveau des obstacles et des terrains difficiles. On peut normaliser le vecteur \vec{g} ou on peut modifier la formule par :

$$\vec{g} = \sum_{i,j} \frac{1}{h_{ij} - h_{00}} \vec{v}_{ij}$$

(en omettant les termes où $h_{ij} = h_{00}$).

Les vecteurs obtenus par la méthode du gradient représentent mieux les spécificités du terrain que la méthode du minimum. On pourrait raffiner en calculant le gradient via une grille 5×5 entourant chaque case. Ci-dessous à gauche, la direction indiquée par le minimum, à droite la direction opposée du gradient (normalisé).



Approximation et interpolation

L'approximation a pour but de modéliser une situation à l'aide d'une fonction simple. Avec une fonction simple, les calculs sont plus rapides. L'interpolation modélise des données partielles par une fonction. On obtient ainsi une fonction qui permet de prédire des valeurs manquantes.

1. Approximation locale

1.1. Développement limité à l'ordre 1

On considère une fonction $f : I \rightarrow \mathbb{R}$ dérivable autant de fois que nécessaire sur un intervalle $I \subset \mathbb{R}$. Fixons un point $x_0 \in I$. Comme f est continue alors $f(x) \simeq f(x_0)$ pour tout x proche de x_0 . Comment obtenir une meilleure approximation ? On va utiliser la dérivée pour écrire un développement limité à l'ordre de 1 de f autour de x_0 :

$$f(x) \simeq f(x_0) + (x - x_0)f'(x_0)$$

En écrivant $x = x_0 + h$ (avec h proche de 0 lorsque x est proche de x_0), une autre écriture possible est :

$$f(x_0 + h) \simeq f(x_0) + hf'(x_0)$$

Géométriquement cela correspond à approcher le graphe de f autour de x_0 par la tangente à ce graphe en x_0 . Bien sûr, un développement limité ne fournit qu'une approximation locale qui n'a aucune utilité si x est trop éloigné de x_0 .

Exemple.

Considérons $f(x) = \frac{1}{\sqrt{x}}$ autour de $x_0 = 1$. On a $f'(x) = -\frac{1}{2x\sqrt{x}}$. Ainsi $f(x_0) = 1$ et $f'(x_0) = -\frac{1}{2}$. Le DL de f d'ordre 1 autour de $x_0 = 1$ est :

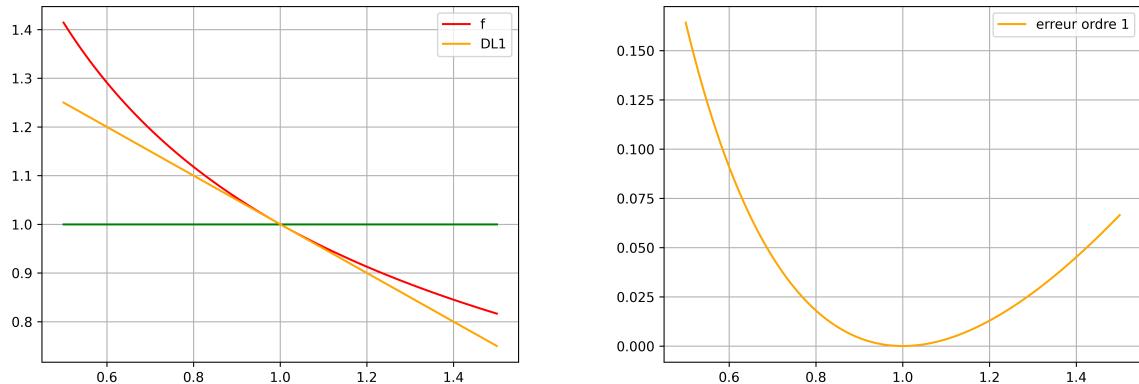
$$\frac{1}{\sqrt{x}} \simeq 1 - \frac{1}{2}(x - 1)$$

ou encore :

$$\frac{1}{\sqrt{1+h}} \simeq 1 - \frac{1}{2}h.$$

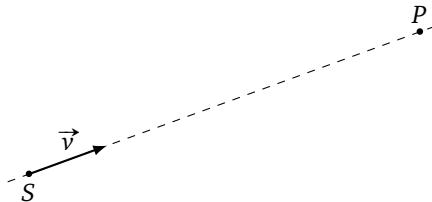
Combien vaut $\frac{1}{\sqrt{1.1}}$? Avec $h = 0.1$, on obtient $\frac{1}{\sqrt{1.1}} \simeq 0.95$. Ce qui est proche de la valeur exacte $\frac{1}{\sqrt{1.1}} = 0.953463\dots$

Sur le dessin de gauche ci-dessous est tracée la courbe de f , ainsi que son DL d'ordre 1 (c'est donc la tangente à la courbe en $x_0 = 1$). Sur la figure de droite on mesure l'erreur commise par le DL d'ordre 1, c'est-à-dire le graphe de $f(x) - (1 - \frac{1}{2}(x - 1))$.



Normalisation. Le calcul de $\frac{1}{\sqrt{x}}$ est particulièrement important car il est utilisé dans la normalisation d'un vecteur. Par exemple : comment trouver le point P situé à une distance t du point S dans la direction du vecteur \vec{v} ? C'est le point :

$$P = S + t \frac{\vec{v}}{\|\vec{v}\|}$$



Si $\vec{v} = (v_x, v_y)$ alors $\frac{\vec{v}}{\|\vec{v}\|} = \frac{1}{\sqrt{v_x^2 + v_y^2}}(v_x, v_y)$. Les calculs de $\frac{1}{\sqrt{x}}$ sont désormais directement implémentés dans les processeurs graphiques. Noter que notre approximation par un DL d'ordre 1 n'est valide que pour des vecteurs qui sont déjà proches d'une norme 1. Une autre technique consiste à appliquer la méthode de Newton, également basée sur la dérivée ; avec un choix judicieux de la valeur initiale, une seule itération peut suffire (« astuce de Carmack »).

1.2. Développement limité à l'ordre n

Voici la formule du DL de f en x_0 à l'ordre n :

$$f(x) \simeq f(x_0) + f'(x_0)(x - x_0) + \frac{f''(x_0)}{2!}(x - x_0)^2 + \cdots + \frac{f^{(n)}(x_0)}{n!}(x - x_0)^n$$

Ce que l'on peut récrire :

$$f(x_0 + h) \simeq f(x_0) + f'(x_0)h + \frac{f''(x_0)}{2!}h^2 + \cdots + \frac{f^{(n)}(x_0)}{n!}h^n$$

c'est-à-dire :

$$f(x_0 + h) \simeq \sum_{k=0}^n \frac{f^{(k)}(x_0)}{k!} h^k.$$

Géométriquement un DL à l'ordre 2 correspond à approcher le graphe de f par une parabole autour de x_0 . Un DL à l'ordre n correspond à approcher le graphe de f par une courbe polynomiale de degré n autour de x_0 . Plus n est grand, plus la courbe polynomiale est proche du graphe de f , mais toujours seulement autour de x_0 .

Exemple.

Considérons $f(x) = \frac{1}{\sqrt{x}} = x^{-\frac{1}{2}}$ autour de $x_0 = 1$. On a $f'(x) = -\frac{1}{2}x^{-\frac{3}{2}}$ et $f''(x) = \frac{3}{4}x^{-\frac{5}{2}}$ et $f'''(x) = -\frac{15}{8}x^{-\frac{7}{2}}$ ce qui permet de calculer les coefficients $\frac{f^{(k)}(x_0)}{k!}$. Les DL à l'ordre 1, 2 et 3 donnent donc :

$$\begin{aligned}\frac{1}{\sqrt{1+h}} &\simeq 1 - \frac{1}{2}h \\ \frac{1}{\sqrt{1+h}} &\simeq 1 - \frac{1}{2}h + \frac{3}{8}h^2 \\ \frac{1}{\sqrt{1+h}} &\simeq 1 - \frac{1}{2}h + \frac{3}{8}h^2 - \frac{5}{16}h^3\end{aligned}$$

Avec $h = 0.1$ on obtient $\frac{1}{\sqrt{1.1}} \simeq 0.953437$ via le DL à l'ordre 3, ce qui donne 4 chiffres exacts après la virgule.

Exemple.

On souhaite approcher $f(x) = \sin(x)$ par un polynôme sur l'intervalle $[0, \frac{\pi}{2}]$.

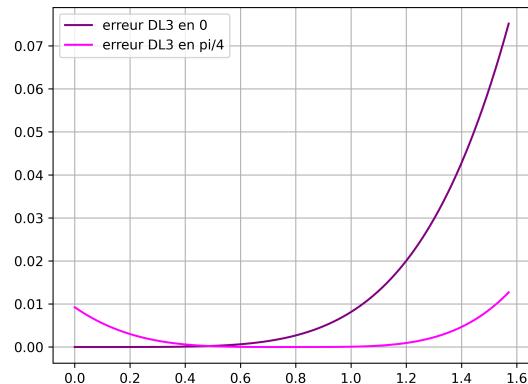
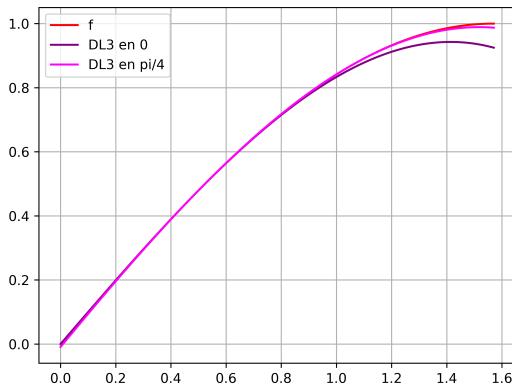
DL en 0. En posant $x_0 = 0$, on a $f(0) = 0$, $f'(0) = 1$, $f''(0) = 0$ et $f'''(0) = -1$. Le DL à l'ordre 3 en 0 est donc :

$$\sin(x) \simeq x - \frac{1}{6}x^3$$

DL en $\frac{\pi}{4}$. En posant $x_0 = \frac{\pi}{4}$, on calcule $f(\frac{\pi}{4}) = \frac{1}{\sqrt{2}}$, $f'(\frac{\pi}{4}) = \frac{1}{\sqrt{2}}$, ... Le DL à l'ordre 3 en $\frac{\pi}{4}$ est donc :

$$\sin(x) \simeq \frac{1}{\sqrt{2}} + \frac{1}{\sqrt{2}} \left(x - \frac{\pi}{4} \right) - \frac{1}{2\sqrt{2}} \left(x - \frac{\pi}{4} \right)^2 - \frac{1}{6\sqrt{2}} \left(x - \frac{\pi}{4} \right)^3$$

Comparaison. Sur $[0, \frac{\pi}{2}]$ le DL en $\frac{\pi}{4}$ est plus précis que le DL en 0 car l'approximation est centrée sur le milieu de l'intervalle.



Temps de calculs. Si on n'a pas besoin d'une grande précision mais d'une grande vitesse de calcul, remplacer le calcul de $\sin(x)$ par un DL à l'ordre 3 est une bonne idée.

Voici les premiers termes des développements limités de fonctions usuelles autour de $x_0 = 0$:

$$\begin{aligned}\exp(x) &\simeq 1 + x + \frac{x^2}{2!} + \frac{x^3}{3!} + \frac{x^4}{4!} + \dots \\ \sqrt{1+x} &\simeq 1 + \frac{x}{2} - \frac{x^2}{8} + \frac{x^3}{16} + \dots \\ \sin(x) &\simeq x - \frac{x^3}{3!} + \frac{x^5}{5!} - \frac{x^7}{7!} + \dots \\ \cos(x) &\simeq 1 - \frac{x^2}{2!} + \frac{x^4}{4!} - \frac{x^6}{6!} + \dots \\ \tan(x) &\simeq x + \frac{x^3}{3} + \frac{2x^5}{15} + \dots \\ \ln(1+x) &\simeq x - \frac{x^2}{2} + \frac{x^3}{3} - \frac{x^4}{4} + \dots\end{aligned}$$

1.3. Deux variables

Soit $f : \mathbb{R}^2 \rightarrow \mathbb{R}$ une fonction à deux variables. Le DL à l'ordre 1 de f en (x_0, y_0) est :

$$f(x_0 + h, y_0 + k) \simeq f(x_0, y_0) + h \frac{\partial f}{\partial x}(x_0, y_0) + k \frac{\partial f}{\partial y}(x_0, y_0)$$

Géométriquement le graphe de f (qui est ici une surface) est approché par le plan tangent à ce graphe en (x_0, y_0) .

Exemple.

La luminosité perçue dépend de l'intensité I_0 de la source et de la distance R_0 à cette source selon la formule

$$L = \frac{I_0}{R_0^2}$$

Comment évolue cette luminosité lorsque l'intensité ou la distance change ? Le DL à l'ordre 1 de $f(x, y) = \frac{x}{y^2}$ en (x_0, y_0) est :

$$f(x_0 + h, y_0 + k) \simeq \frac{x_0}{y_0^2} + h \frac{1}{y_0^2} - 2k \frac{x_0}{y_0^3}$$

Ce qui pour notre problème donne :

$$L(I_0 + \Delta I, R_0 + \Delta R) \simeq \frac{I_0}{R_0^2} + \Delta I \frac{1}{R_0^2} - 2\Delta R \frac{I_0}{R_0^3}$$

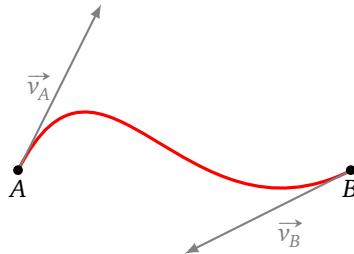
Application numérique avec $(I_0, R_0) = (100, 10)$, qui donne une luminosité initiale $L(I_0, R_0) = 1$, et $\Delta I = 10$ et $\Delta R = 1$:

$$L(110, 11) \simeq 1 + 0.1 - 0.2 = 0.9.$$

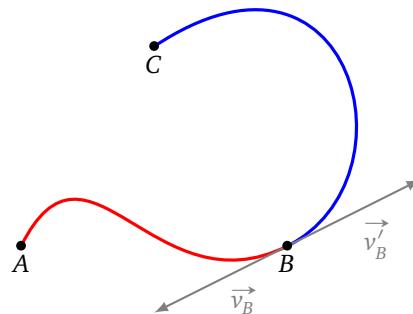
2. Courbes de Bézier

2.1. Cubique de Bézier

Expliquons intuitivement les contraintes pour tracer une courbe de Bézier dans l'un des cas le plus simple. On se donne deux points A et B , ainsi que deux vecteurs : un vecteur \vec{v}_A issu de A et un vecteur \vec{v}_B issu de B . La cubique de Bézier est une courbe qui passe par A et B et dont les tangentes sont données par les vecteurs \vec{v}_A et \vec{v}_B . Plus les vecteurs sont grands, plus la courbe reste tangente à ces vecteurs.



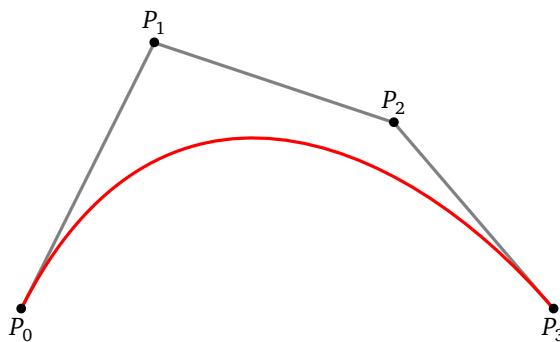
Comme on contrôle le vecteur tangent aux extrémités, on peut en recoller deux ensembles de façon à obtenir une courbe globale lisse.



Revenons à la définition mathématique de la cubique de Bézier. Soient $P_0, P_1, P_2, P_3 \in \mathbb{R}^2$ quatre points du plan. On appelle **cubique de Bézier** le graphe de la fonction $\gamma : [0, 1] \rightarrow \mathbb{R}^2$ définie par :

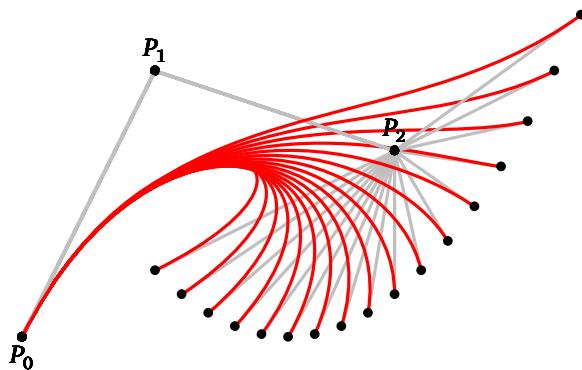
$$\boxed{\gamma(t) = (1-t)^3 P_0 + 3(1-t)^2 t P_1 + 3(1-t)t^2 P_2 + t^3 P_3}$$

Cette courbe part de P_0 suivant le vecteur tangent $\overrightarrow{P_0P_1}$ puis arrive à P_3 suivant le vecteur tangent $\overrightarrow{P_2P_3}$ (ainsi P_0 joue le rôle de A , P_3 celui de B et $\vec{v}_A = \overrightarrow{P_0P_1}$ et $\vec{v}_B = \overrightarrow{P_2P_3}$). Attention la courbe ne passe pas par les points P_1 et P_2 !

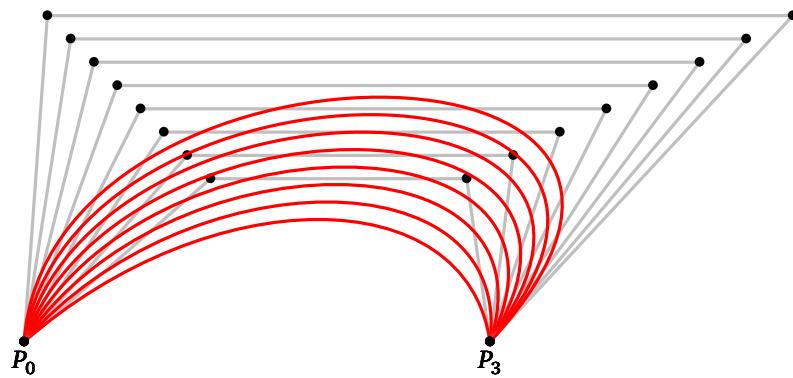


Il faut comprendre l'addition de points comme une addition de vecteurs de \mathbb{R}^2 : $aP + bQ$ a pour coordonnées $(ax_P + bx_Q, ay_P + by_Q)$. Ainsi pour $t = 0$ on a $\gamma(0) = P_0$ et pour $t = 1$ on a $\gamma(1) = P_3$.

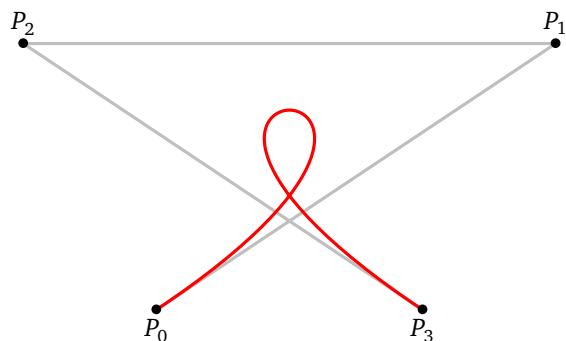
Voici différents exemples de courbes de Bézier. Tout d'abord si on change la position du point d'arrivée P_3 seulement.



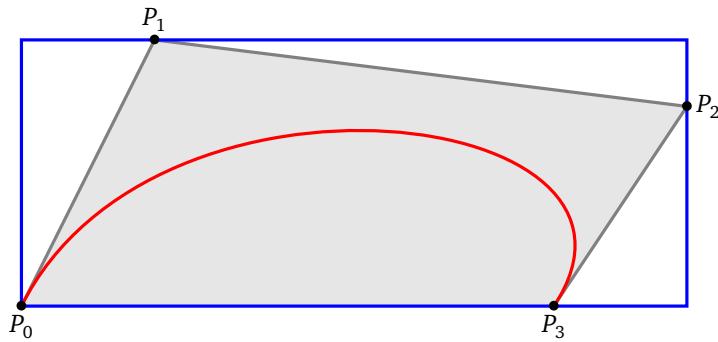
Ci-dessous on change seulement les vecteurs tangents.



La courbe de Bézier peut parfois avoir des formes surprenantes, par exemple elle peut se recouper elle-même.



Bounding box. Noter que la courbe de Bézier est toujours contenue dans le quadrilatère convexe de sommets P_0, P_1, P_2, P_3 . Cela permet d'obtenir un rectangle appelé *bounding box* (ou boîte englobante) de cette courbe. Attention, ce rectangle n'est pas nécessairement le rectangle minimal pour la courbe de Bézier.



2.2. Vecteur tangent

Soit $\gamma : [0, 1] \rightarrow \mathbb{R}^2$ une courbe de Bézier cubique donnée par :

$$\gamma(t) = (1-t)^3 P_0 + 3(1-t)^2 t P_1 + 3(1-t)t^2 P_2 + t^3 P_3$$

Alors le vecteur tangent à la courbe au point $\gamma(t)$ est donné par :

$$\boxed{\gamma'(t) = 3(1-t)^2 \overrightarrow{P_0 P_1} + 6(1-t)t \overrightarrow{P_1 P_2} + 3t^2 \overrightarrow{P_2 P_3}}$$

Voici la preuve :

$$\begin{aligned} \gamma'(t) &= \frac{d}{dt} ((1-t)^3 P_0 + 3(1-t)^2 t P_1 + 3(1-t)t^2 P_2 + t^3 P_3) \\ &= -3(1-t)^2 P_0 - 6(1-t)t P_1 + 3(1-t)^2 P_1 + 6(1-t)t P_2 - 3t^2 P_2 + 3t^2 P_3 \\ &= 3(1-t)^2 (P_1 - P_0) + 6(1-t)t (P_2 - P_1) + 3t^2 (P_3 - P_2) \\ &= 3(1-t)^2 \overrightarrow{P_0 P_1} + 6(1-t)t \overrightarrow{P_1 P_2} + 3t^2 \overrightarrow{P_2 P_3} \end{aligned}$$

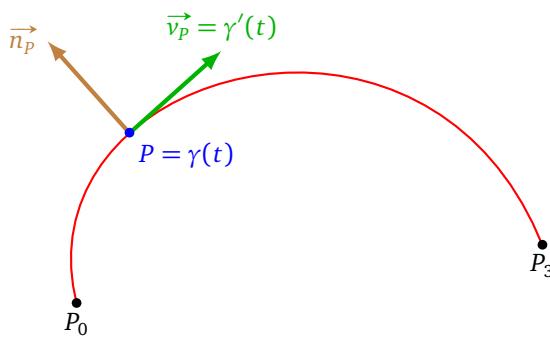
Ainsi le vecteur tangent initial (au point $\gamma(0) = P_0$) est :

$$\gamma'(0) = 3 \overrightarrow{P_0 P_1}.$$

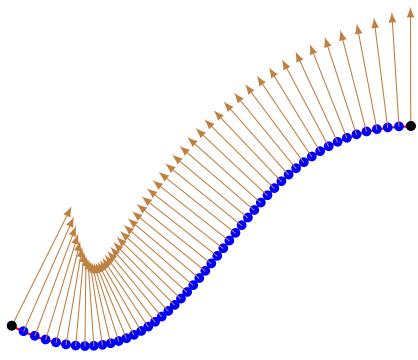
Celui à la fin (au point $\gamma(1) = P_3$) est :

$$\gamma'(1) = -3 \overrightarrow{P_3 P_2}.$$

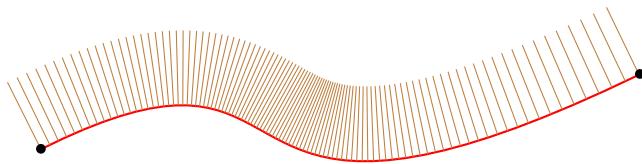
Ainsi la courbe est bien tangente au départ et à l'arrivée aux vecteurs $\overrightarrow{P_0 P_1}$ et $\overrightarrow{P_3 P_2}$.



Une fois que l'on connaît le vecteur tangent $\vec{v} = (a, b)$ en un point de la courbe, on calcule simplement le vecteur normal $\vec{n} = (-b, a)$, qui est un vecteur perpendiculaire à la tangente à la courbe.



Il est utile de rendre ces vecteurs normaux unitaires.



Connaître les vecteurs tangents et normaux unitaires permet de dessiner le long d'une courbe, tracer une courbe parallèle (comme des rails, mais attention le second rail n'est pas un translaté du premier) ou bien de dessiner ou d'écrire du texte le long d'une courbe.

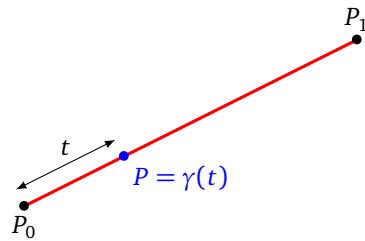
Si j'ai pu voir plus loin, c'est que je me tenais sur les épaules de géants. Isaac Newton

2.3. Construction récursive

Pierre Bézier (1910-1999) était ingénieur chez Renault et cherchait des courbes simples à calculer et à dessiner. C'est Paul de Casteljau (1930-2022, ingénieur chez Citroën !) qui propose un algorithme pour construire facilement les courbes de Bézier.

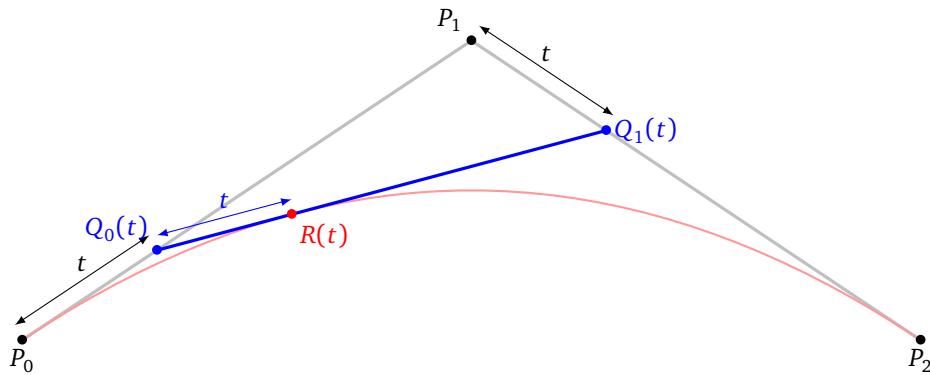
Avec deux points. Soient $P_0, P_1 \in \mathbb{R}^2$ deux points du plan. La courbe de Bézier est alors simplement le segment $[P_0, P_1]$ paramétré par :

$$\gamma(t) = (1-t)P_0 + tP_1$$

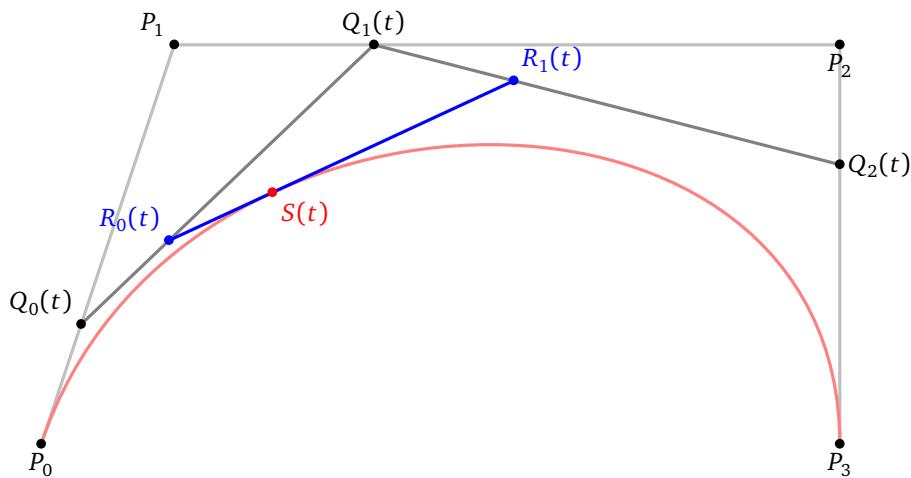


Nous parlerons de **paramétrisation linéaire** (abrégé en *lerp* pour *linear interpolation*). Cela signifie que $\gamma(0) = P_0$ et $\gamma(1) = P_1$ et $\gamma(t)$ est à une distance t de P_0 et à une distance $1 - t$ de P_1 (si on suppose que la distance P_0P_1 vaut 1).

Avec trois points. Soient $P_0, P_1, P_2 \in \mathbb{R}^2$ trois points du plan. Pour tracer la courbe de Bézier, on calcule d'une part la paramétrisation linéaire $Q_0(t)$ du segment $[P_0, P_1]$ et d'autre part la paramétrisation linéaire $Q_1(t)$ du segment $[P_1, P_2]$. Pour chaque t , on obtient un segment $[Q_0(t), Q_1(t)]$. On calcule alors la paramétrisation linéaire $R(t)$ de ce segment. Ce point $R(t)$ est le point de la courbe de Bézier au paramètre t .



Avec quatre points. On part de quatre points P_0, P_1, P_2, P_3 , on calcule trois paramétrisations linéaires correspondant aux trois segments, on obtient ainsi trois points $Q_0(t), Q_1(t), Q_2(t)$, puis on calcule deux paramétrisations linéaires $R_0(t), R_1(t)$ correspondant aux deux segments $[Q_0(t), Q_1(t)], [Q_1(t), Q_2(t)]$, finalement le point final $S(t)$ de la courbe de Bézier est celui de la paramétrisation linéaire de $[R_0(t), R_1(t)]$. Noter que cet algorithme est récursif : on calcule le point d'une courbe de Bézier de n points en calculant des points de plusieurs courbes de Bézier de $n - 1$ points. Il faut bien noter que le paramètre $t \in [0, 1]$ est le même pour toutes les courbes de Bézier intermédiaires.



2.4. Cas général

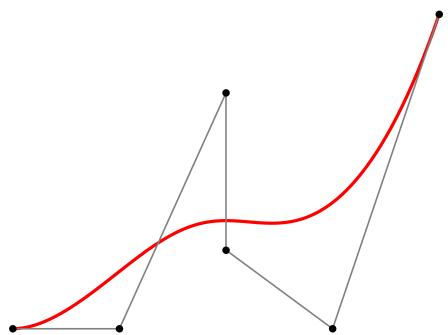
Les formules et constructions précédentes se généralisent pour obtenir la courbe de Bézier pour $n+1$ points P_0, \dots, P_n .

Le **polynôme de Bernstein** de degré n est défini par :

$$b_{i,n}(t) = \binom{n}{i} t^i (1-t)^{n-i}$$

où $\binom{n}{i} = \frac{n!}{i!(n-i)!}$ est le coefficient binomial « i parmi n ». La **courbe de Bézier** associée à P_0, \dots, P_n est alors paramétrée par :

$$\gamma(t) = \sum_{i=0}^n b_{i,n}(t) P_i$$

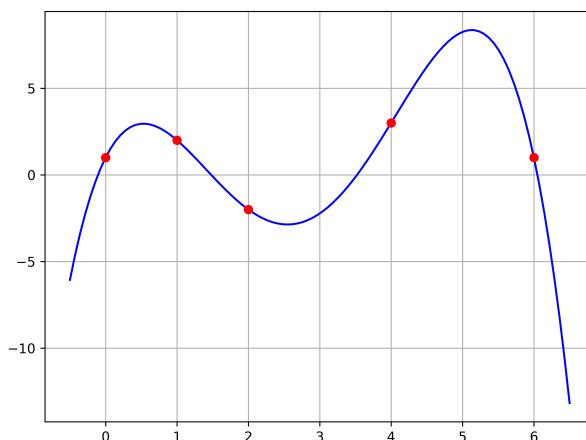


3. Approximation globale

3.1. Interpolation de Lagrange

Le problème est le suivant : on nous donne $n+1$ points dans le plan et on cherche une courbe passant exactement par tous ces points.

Précisons le problème : soient $(a_0, b_0), \dots, (a_n, b_n)$, $n+1$ points du plan avec $a_0 < a_1 < \dots < a_n$. On cherche un polynôme $P(X)$ de degré $\leq n$ tel que $P(a_i) = b_i$ pour tout $i = 0, \dots, n$.



Proposition 1.

Si on note le polynôme de Lagrange élémentaire

$$L_i(X) = \prod_{\substack{j=0 \\ j \neq i}}^n \frac{X - a_j}{a_i - a_j}$$

alors l'unique polynôme de degré $\leq n$ vérifiant $P(a_i) = b_i$ pour tout $i = 0, \dots, n$, est :

$$P(X) = \sum_{i=0}^n b_i L_i(X)$$

Pour la preuve le point clé est de remarquer que pour le polynôme élémentaire $L_i(X)$, on a :

$$L_i(a_i) = 1 \quad \text{et} \quad L_i(a_j) = 0 \text{ pour tout } j \neq i.$$

Démonstration. Montrons d'abord que le polynôme $P(X)$ (qui est bien de degré n) convient. En effet

$$P(a_i) = \sum_{j=0}^n b_j L_j(a_i) = b_i L_i(a_i) = b_i$$

car pour $j \neq i$, on a $L_j(a_i) = 0$.

Soit maintenant $Q(X)$ un autre polynôme de degré $\leq n$ vérifiant $Q(a_i) = b_i$ pour tout $i = 0, \dots, n$. On a alors $(P - Q)(a_i) = 0$ pour tout $i = 0, \dots, n$. Donc le polynôme $P(X) - Q(X)$ est un polynôme de degré $\leq n$ ayant $n + 1$ racines distinctes, c'est donc le polynôme nul. D'où $P(X) = Q(X)$. \square

Exemple.

Considérons $n = 2$ et les 3 points $(a_0, b_0), (a_1, b_1)$ et (a_2, b_2) . Trouvons l'équation de l'unique parabole passant par ces trois points. Tout d'abord :

$$L_0(X) = \frac{(X - a_1)(X - a_2)}{(a_0 - a_1)(a_0 - a_2)} \quad L_1(X) = \frac{(X - a_0)(X - a_2)}{(a_1 - a_0)(a_1 - a_2)} \quad L_2(X) = \frac{(X - a_0)(X - a_1)}{(a_2 - a_0)(a_2 - a_1)}$$

et donc

$$P(X) = b_0 L_0(X) + b_1 L_1(X) + b_2 L_2(X).$$

Par exemple pour les points $(0, 2), (1, 1)$ et $(3, 4)$, on a :

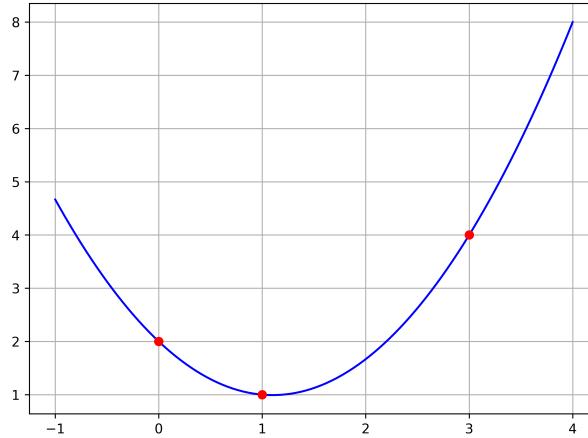
$$L_0(X) = \frac{(X - 1)(X - 3)}{(0 - 1)(0 - 3)} = \frac{1}{3}(X^2 - 4X + 3)$$

$$L_1(X) = \frac{(X - 0)(X - 3)}{(1 - 0)(1 - 3)} = -\frac{1}{2}(X^2 - 3X)$$

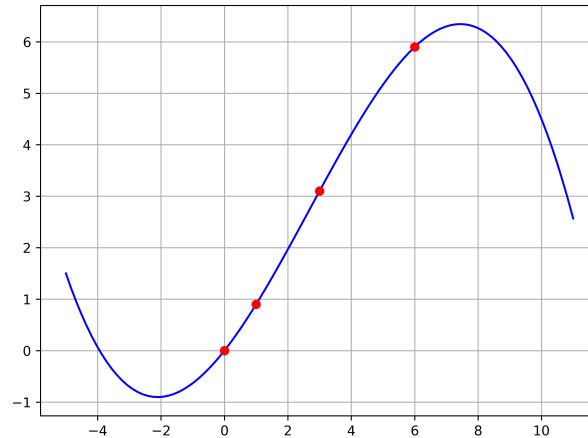
$$L_2(X) = \frac{(X - 0)(X - 1)}{(3 - 0)(3 - 1)} = \frac{1}{6}(X^2 - X)$$

et donc

$$P(X) = \frac{2}{3}(X^2 - 4X + 3) - \frac{1}{2}(X^2 - 3X) + \frac{4}{6}(X^2 - X) = \frac{1}{6}(5X^2 - 11X + 12)$$

**Exemple.**

Voici un exemple avec 4 points ($n = 3$).



On remarque que même si ces quatre points sont presque alignés la courbe obtenue est très éloignée d'une courbe affine. On peut se demander si imposer de passer exactement par les points n'est pas une contrainte trop forte.

3.2. Polynômes de Tchebychev

Décrivons deux façons de définir les polynômes de Tchebychev $T_n(X)$.

Définition trigonométrique. Le polynôme $T_n(X)$ est défini par la relation :

$$T_n(X) = \cos(n \arccos X)$$

Autrement dit :

$$T_n(\cos \theta) = \cos(n\theta)$$

Définition par récurrence. Le polynôme $T_n(X)$ est aussi défini par la relation de récurrence :

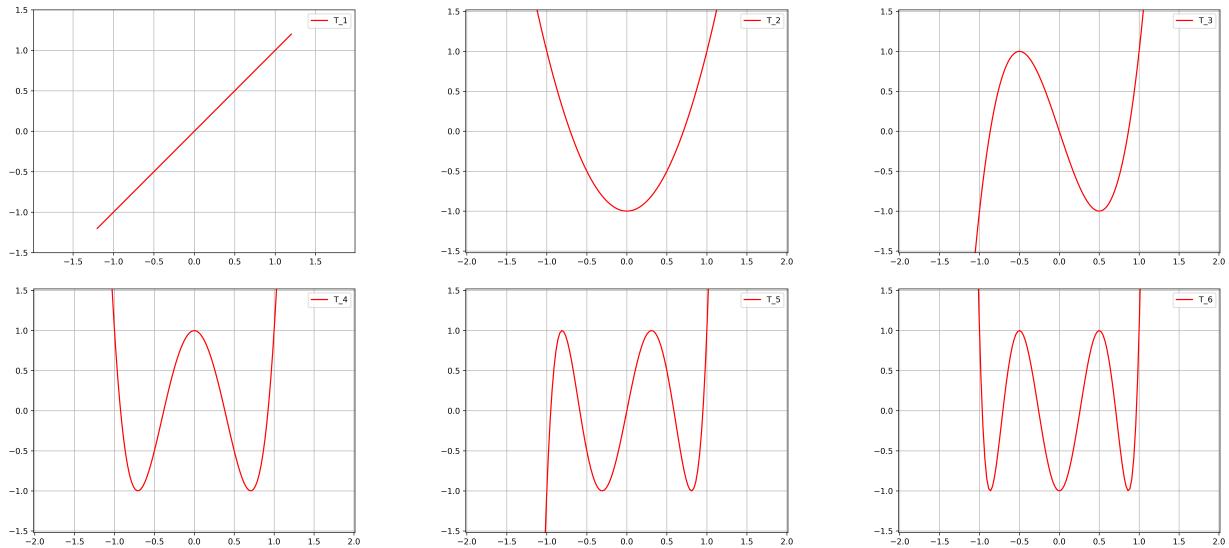
$$T_0(X) = 1 \quad T_1(X) = X \quad T_{n+1}(X) = 2XT_n(X) - T_{n-1}(X) \quad \text{pour } n \geq 1$$

Voici la liste des premiers polynômes de Tchebychev :

$$T_0(X) = 1 \quad T_1(X) = X \quad T_2(X) = 2X^2 - 1$$

$$T_3(X) = 4X^3 - 3X \quad T_4(X) = 8X^4 - 8X^2 + 1 \quad T_5(X) = 16X^5 - 20X^3 + 5X$$

Voici les graphes de ces polynômes $T_n(X)$ pour $n = 1, \dots, 6$.



Voici quelques propriétés immédiates des polynômes de Tchebychev :

Proposition 2.

- $T_n(X)$ est un polynôme de degré n de coefficient dominant 2^{n-1} (pour tout $n \geq 1$).
- Le polynôme $T_n(X)$ admet n racines distinctes dans $[-1, 1]$ qui sont :

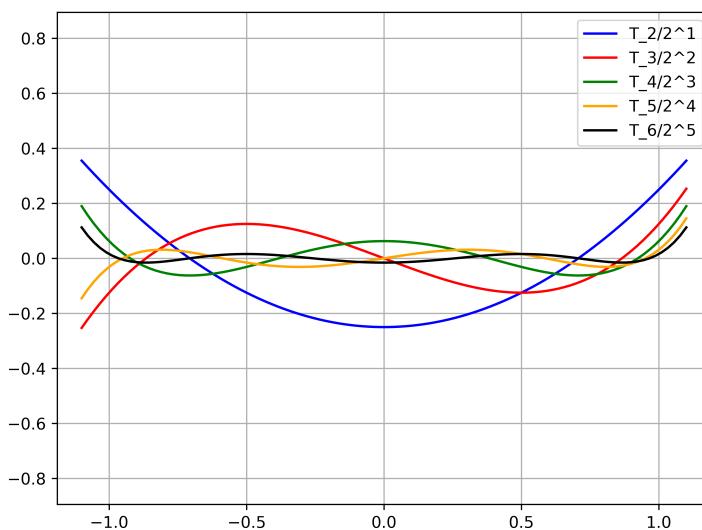
$$\omega_k = \cos\left(\frac{2k+1}{2n}\pi\right) \quad k = 0, \dots, n-1.$$

La propriété qui nous concerne davantage est la suivante :

Proposition 3.

Le polynôme $\frac{1}{2^{n-1}}T_n(X)$ est le polynôme qui approche au mieux la fonction nulle sur l'intervalle $[-1, 1]$ parmi tous les polynômes de degré n et de coefficient dominant 1.

La reformulation mathématique de cette proposition est la suivante : soit $\|f\|_\infty = \max_{-1 \leq x \leq 1} |f(x)|$, la norme infinie d'une fonction f sur l'intervalle $[-1, 1]$. Alors $\|T_n\|_\infty = \frac{1}{2^{n-1}}$ est la plus petite norme possible pour un polynôme de degré n et de coefficient dominant 1.



Évidemment approcher la fonction nulle a peu d'intérêt en soi mais prouve que les polynômes de Tchebychev permettent d'approcher efficacement une fonction sur un intervalle tout entier et pas seulement en quelques points. Les polynômes de Tchebychev permettent en fait d'approcher n'importe quelle fonction sur un intervalle $[a, b]$ par un polynôme de degré n (de façon analogue aux séries de Fourier). Nous énonçons le résultat de manière informelle.

Soit $f : [-1, 1] \rightarrow \mathbb{R}$ une fonction continue et dérivable. Il existe des coefficients $c_i \in \mathbb{R}$ tels que $P(X) = \sum_{i=0}^{n-1} c_i T_i(X)$ est un polynôme de degré $n - 1$ qui approche correctement f sur l'intervalle $[-1, 1]$:

$$f(x) \simeq \sum_{i=0}^{n-1} c_i T_i(x) \quad \text{pour tout } x \in [-1, 1]$$

où les coefficients se calculent par les formules :

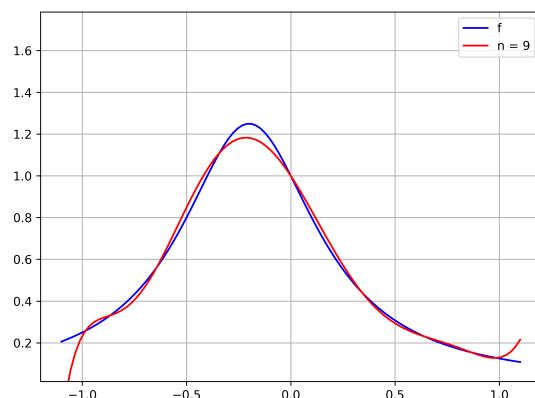
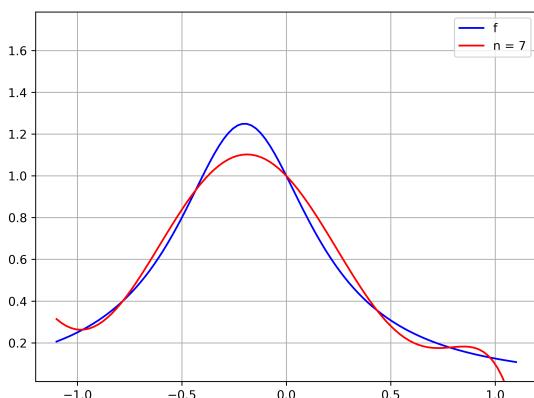
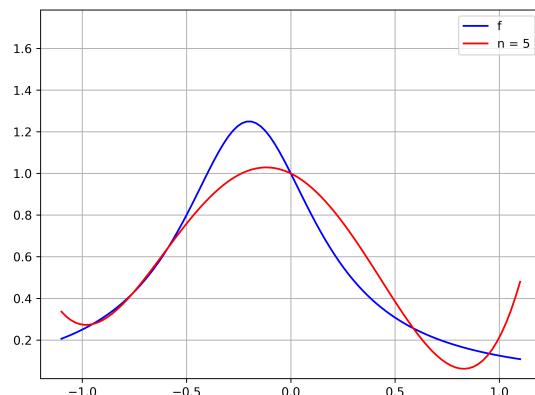
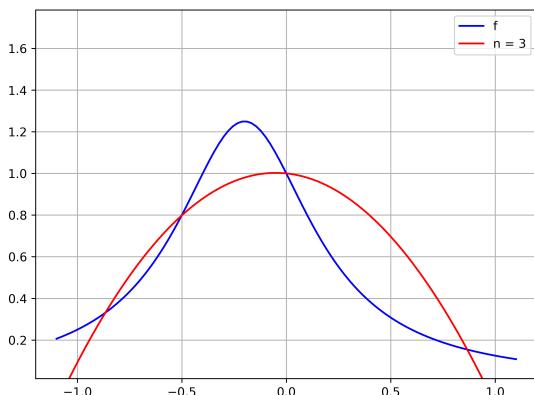
$$c_0 = \frac{1}{n} \sum_{k=0}^{n-1} f(\omega_k) \quad \text{et} \quad c_i = \frac{2}{n} \sum_{k=0}^{n-1} f(\omega_k) T_i(\omega_k) \quad \text{pour } i = 1, \dots, n-1$$

et où les ω_k sont les n racines de $T_n(X)$.

L'erreur que l'on cherche à minimiser est $\|f - P\|_\infty$. Les polynômes de Tchebychev ne fournissent pas une approximation optimale mais une approximation qui est suffisamment bonne pour être utilisée dans la plupart des cas.

Exemple.

Voici les approximations de la fonction $f(x) = \frac{1}{1+2x+5x^2}$ par des séries de Tchebychev pour les ordres $n = 3, 5, 7, 9$.



Exemple.

On souhaite approcher $g(x) = \sin(x)$ sur l'intervalle $[0, \frac{\pi}{2}]$. On commence par transformer la fonction g définie sur $[0, \frac{\pi}{2}]$ en une fonction f définie sur $[-1, 1]$ par :

$$f(x) = g\left(\frac{x+1}{2} \frac{\pi}{2}\right) = \sin\left((x+1)\frac{\pi}{4}\right)$$

Calculons le développement en série de Tchebychev de f pour $n = 3$.

- Tout d'abord, on rappelle :

$$T_0(X) = 1 \quad T_1(X) = X \quad T_2(X) = 2X^2 - 1 \quad T_3(X) = 4X^3 - 3X$$

- Les racines de $T_3(X)$ sont :

$$\omega_0 = \cos\left(\frac{1}{6}\pi\right) = \frac{\sqrt{3}}{2} \quad \omega_1 = \cos\left(\frac{3}{6}\pi\right) = 0 \quad \omega_2 = \cos\left(\frac{5}{6}\pi\right) = -\frac{\sqrt{3}}{2}$$

- Les coefficients c_i sont :

$$c_0 = \frac{1}{3}(f(\omega_0) + f(\omega_1) + f(\omega_2)) \simeq 0.602202$$

$$c_1 = \frac{2}{3}(f(\omega_0)T_1(\omega_0) + f(\omega_1)T_1(\omega_1) + f(\omega_2)T_1(\omega_2)) \simeq 0.513518$$

$$c_2 = \frac{2}{3}(f(\omega_0)T_2(\omega_0) + f(\omega_1)T_2(\omega_1) + f(\omega_2)T_2(\omega_2)) \simeq -0.104905$$

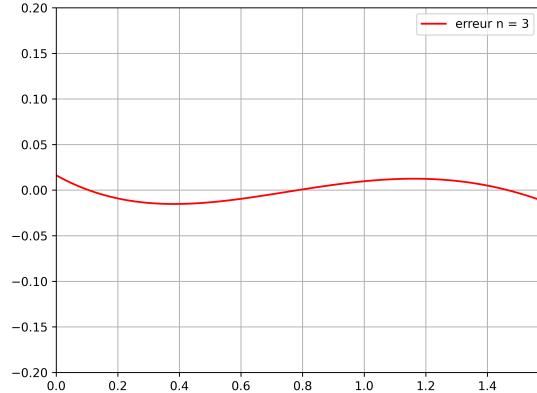
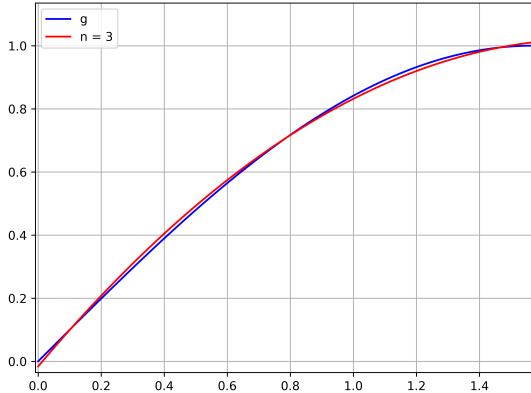
- Ainsi le développement en série de Tchebychev de f pour $n = 3$ est :

$$f(x) \simeq P(x) = c_0 T_0(x) + c_1 T_1(x) + c_2 T_2(x) = (c_0 - c_2) + c_1 x + 2c_2 x^2$$

- Revenons à notre problème initial d'approcher $g(x) = \sin(x)$ sur l'intervalle $[0, \frac{\pi}{2}]$. On effectue la transformation inverse sur $P(X)$

$$Q(X) = P\left(\frac{4X}{\pi} - 1\right)$$

- Voici le graphe obtenu pour $Q(x)$ comparé avec celui de $g(x)$ (figure de gauche). L'erreur commise par cette approximation sur $[0, \frac{\pi}{2}]$ est partout inférieure à 0.02 (figure de droite).



3.3. Approximants de Padé

Au lieu d'approcher une fonction sur un intervalle par des polynômes on peut aussi l'approcher par des fractions rationnelles de la forme :

$$\frac{R(x)}{Q(x)} = \frac{a_0 + a_1 x + \cdots + a_n x^n}{b_0 + b_1 x + \cdots + b_m x^m}$$

où $R(x)$ et $Q(x)$ sont des polynômes de degrés respectifs n et m . Avec des polynômes de degrés plus petits on obtient une approximation équivalente à celle obtenue par un polynôme $P(X)$ de degré $n + m$.

Exemple.

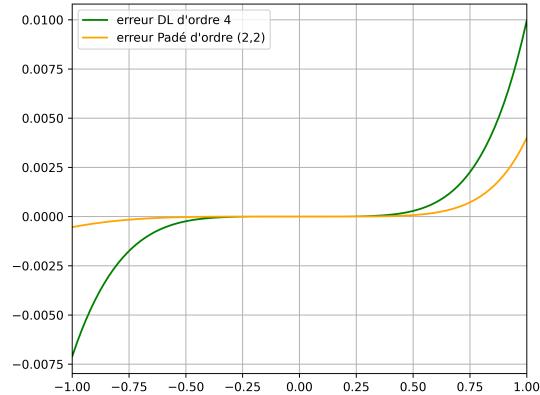
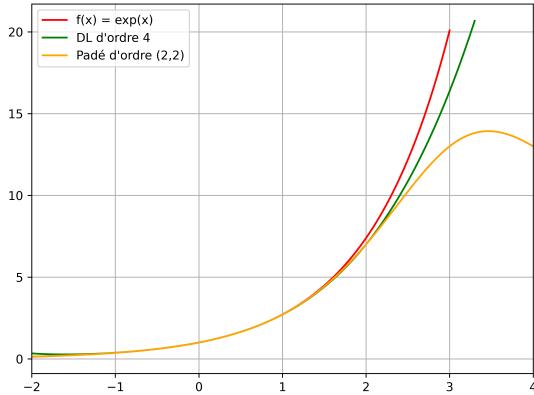
Soit $f(x) = \exp(x)$ à approcher autour de $x_0 = 0$. Le développement limité à l'ordre 4 autour de 0 est :

$$P(x) = 1 + x + \frac{x^2}{2} + \frac{x^3}{6} + \frac{x^4}{24}.$$

L'approximant de Padé de bidegré $(n, m) = (2, 2)$ est :

$$\frac{R(x)}{Q(x)} = \frac{12 + 6x + x^2}{12 - 6x + x^2}.$$

Les comportements de $P(x)$ et de $\frac{R(x)}{Q(x)}$ autour de $x_0 = 0$ sont quasi-identiques mais les approximants de Padé peuvent avoir un comportement plus adapté loin de x_0 .



Équations différentielles

Les équations différentielles apparaissent naturellement dans de nombreux domaines au-delà des mathématiques. Elles permettent de modéliser des phénomènes d'évolution en physique, biologie, économie... Nous expliquons ici comment trouver des solutions approchées de ces équations grâce à des méthodes numériques de discréétisation.

1. Méthode d'Euler

1.1. Qu'est-ce qu'une équation différentielle ?

Une équation différentielle n'est pas comme une équation classique où le but est de trouver une valeur x . Dans une équation différentielle l'inconnue est une fonction $y(x)$ et l'équation fait intervenir la fonction $y(x)$ ainsi que sa dérivée $y'(x)$ (et éventuellement sa dérivée seconde $y''(x)$).

Voici un exemple d'équation différentielle du premier ordre :

$$y'(x) = y(x).$$

Il s'agit donc de trouver une fonction $y(x)$ telle que sa dérivée est égale à la fonction elle-même. Une solution est simplement $y(x) = e^x$. Cette solution n'est pas unique, $y(x) = ke^x$ est aussi une solution, pour toute constante $k \in \mathbb{R}$.

Le principe fondamental de la mécanique, qui relie l'accélération d'un objet aux forces en présence, conduit naturellement à des équations différentielles du second ordre car l'accélération est la dérivée seconde de la position.

Pour certaines équations différentielles il est possible de trouver une solution exacte (on renvoie à un cours classique de mathématiques), mais ce n'est pas toujours possible. Ici nous allons calculer numériquement des solutions approchées des équations différentielles.

1.2. Algorithme

L'idée est simple, si on connaît la valeur de $y(x)$ et de $y'(x)$ en une valeur x_0 alors on peut calculer une valeur approchée de $y(x_0 + h)$ par la formule du développement limité à l'ordre 1 :

$$y(x_0 + h) \simeq y(x_0) + hy'(x_0)$$

On va ensuite calculer, de proche en proche, la valeur de $y(x_0 + 2h)$, $y(x_0 + 3h)$... Nous allons considérer les équations différentielles du premier ordre données par une formule :

$$y'(x) = f(x, y(x))$$

où $f : \mathbb{R}^2 \rightarrow \mathbb{R}$ est une fonction donnée. Cette forme d'équation permet de calculer la dérivée $y'(x_0)$ en utilisant la valeur de $y(x_0)$ et donc par la formule précédente de calculer une valeur approchée de $y(x_0 + h)$.

Par exemple, pour $f(x, y) = xy$, l'équation différentielle est $y'(x) = xy(x)$ (dont en fait les solutions sont $y(x) = ke^{x^2/2}$ ($k \in \mathbb{R}$)). Si on considère $x_0 = 2$ et par exemple on sait que $y(2) = 3$, alors par l'équation différentielle on sait que $y'(2) = x_0 y'(x_0) = 6$. Ainsi $y(2+h) \simeq 3+6h$. Par exemple $y(2.1) \simeq 3+6 \times 0.1 = 3.6$.

L'algorithme est donc le suivant :

Algorithme.

Méthode d'Euler.

Entrées :

- une fonction f définissant une équation différentielle $y'(x) = f(x, y(x))$,
- un intervalle $[a, b]$ et un nombre n de subdivisions,
- une valeur initiale y_0 pour $y(a)$.

Sortie : une liste y_i des valeurs approchées de $y(x_i)$ pour $i = 0, \dots, n$.

- $h = \frac{b-a}{n}$
- $x_0 = a$
- $y_0 = y(a)$
- Pour i variant de 0 à $n-1$:
 - $y_{i+1} = y_i + hf(x_i, y_i)$
 - $x_{i+1} = x_i + h$

1.3. Calculs numériques

Considérons l'équation différentielle

$$y'(x) = \frac{1}{2y(x)}$$

avec la condition initiale $y(1) = 1$.

La solution de ce problème est tout simplement $y(x) = \sqrt{x}$. Nous allons appliquer la méthode d'Euler sur l'intervalle $[1, 2]$ afin d'obtenir une valeur approchée de $y(2) = \sqrt{2}$.

Voici ce que cela donne pour $n = 10$ subdivisions :

i	x_i	y_i
0	1	1
1	1.1	1.05
2	1.2	1.09
3	1.3	1.14
4	1.4	1.18
5	1.5	1.22
6	1.6	1.26
7	1.7	1.30
8	1.8	1.34
9	1.9	1.38
10	2	1.42

On trouve la valeur approchée de $\sqrt{2} \simeq 1.42$. La valeur exacte étant $\sqrt{2} = 1.4142\dots$

Recommençons avec $n = 100$ subdivisions :

i	x_i	y_i
0	1	1
1	1.01	1.0050
2	1.02	1.0099
...		
100	2	1.4148

On obtient donc trois décimales exactes pour $\sqrt{2}$.

1.4. Visualisation

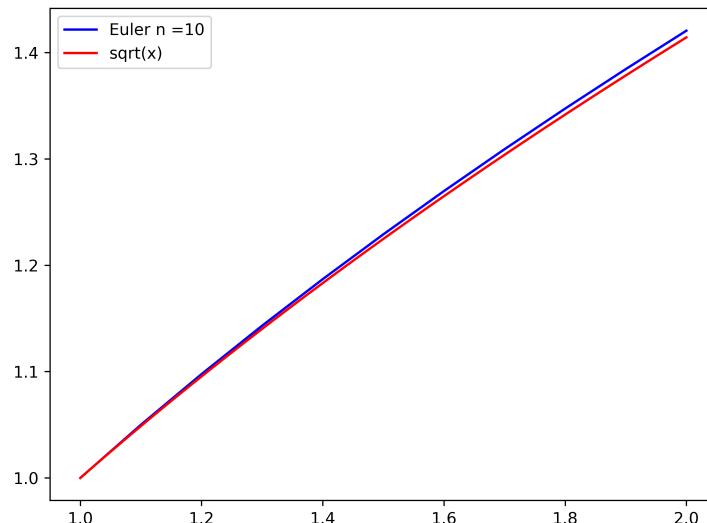
Comparons les graphes de nos solutions approchées avec la solution exacte.

Exemple.

Soit tout d'abord l'équation différentielle avec une condition initiale :

$$y'(x) = \frac{1}{2y(x)} \quad \text{et} \quad y(1) = 1.$$

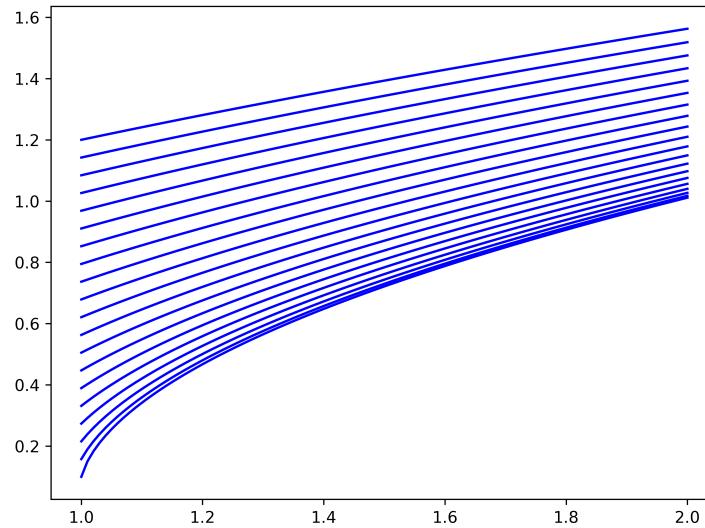
La solution exacte est $y(x) = \sqrt{x}$.



Si on ne fixe pas de condition initiale, l'équation différentielle

$$y'(x) = \frac{1}{2y(x)}$$

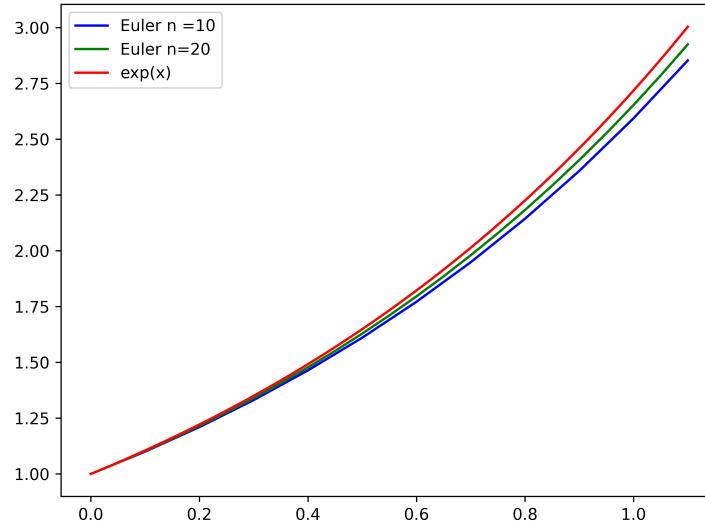
a une infinité de solutions $y(x) = \sqrt{x+c}$, où c est une constante. Dès que l'on fixe une condition initiale $y(x_0) = y_0$ alors la solution devient unique. Pour quelques valeurs initiales, on trace le graphe des solutions ci-dessous.

**Exemple.**

Considérons l'équation différentielle avec une condition initiale :

$$y'(x) = y(x) \quad \text{et} \quad y(1) = 1.$$

On trace le graphe de la solution exacte et des solutions approchées avec $n = 10$ et $n = 20$ subdivisions (pour $n = 100$ le graphe de la solution approchée serait confondu avec le graphe de la solution exacte).



1.5. Vitesse d'un parachutiste

Déterminons la vitesse d'un parachutiste lors de son saut. Le parachutiste est soumis à deux forces : son poids $\vec{P} = m\vec{g}$ et la force de frottement $\vec{F} = -f \|\vec{v}\|^2 \frac{\vec{v}}{\|\vec{v}\|}$ proportionnelle au carré de la vitesse (f est le coefficient de frottement). Le principe fondamental de la mécanique s'écrit :

$$\vec{P} + \vec{F} = m\vec{a}$$

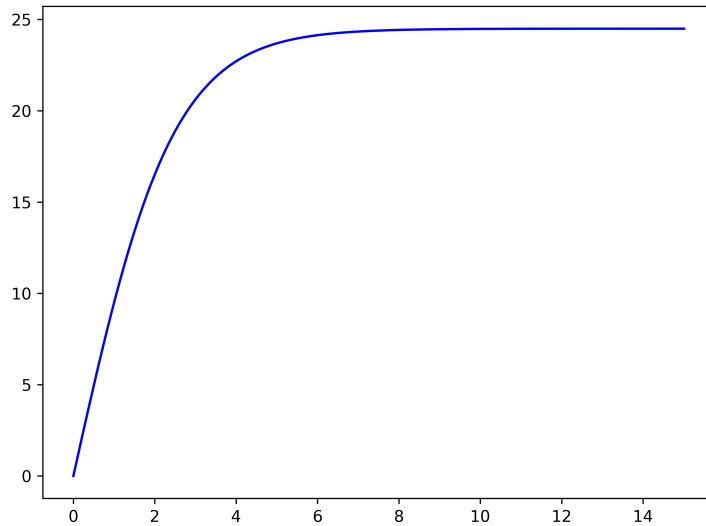
Ce qui conduit à :

$$mg - f v^2(t) = m \frac{dv(t)}{dt}$$

Ainsi l'équation différentielle est :

$$\frac{dv(t)}{dt} = g - \frac{f}{m}v^2(t).$$

Autrement dit avec nos notations précédentes $y'(t) = g - \frac{f}{m}y^2(t)$ où $y(t)$ serait la vitesse du parachutiste. Comme condition initiale on choisit $v(0) = 0$, c'est-à-dire qu'à l'instant initial le parachutiste démarre avec une vitesse nulle. Il n'est pas évident de déterminer la fonction v solution de cette équation différentielle. Cependant la méthode d'Euler nous fournit une solution approchée (dessinée ci-dessous pour $g = 10$, $m = 60$ et $f = 1$).



La solution graphique met en évidence un phénomène vérifié expérimentalement : la vitesse augmente mais est plafonnée par une vitesse limite.

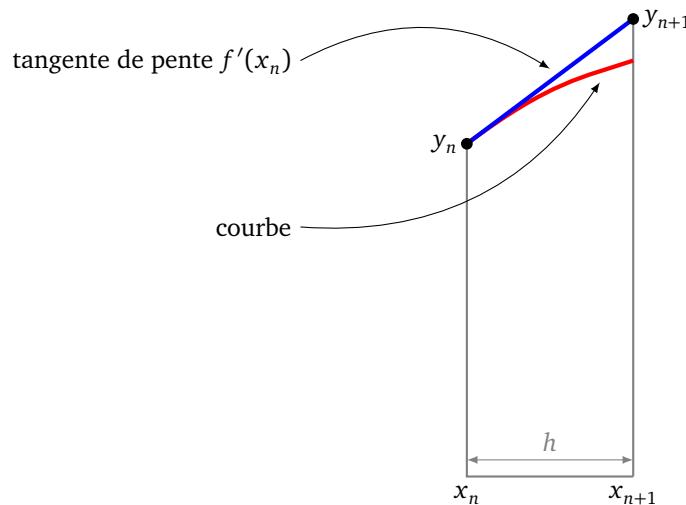
Si on s'intéresse à la position x du parachutiste (c'est-à-dire sa hauteur) cela revient à trouver une primitive de v , ce qui est une forme simple de résolution d'une équation différentielle : $v(t) = \frac{dx(t)}{dt}$. On verra en fin de chapitre de tels exemples.

2. Méthodes de Runge-Kutta

2.1. Retour sur la méthode d'Euler

Expliquons graphiquement un pas de la méthode d'Euler. On se place en x_n , on suppose connue (ou estimée) la valeur y_n de la fonction y en x_n . L'équation différentielle nous donne la valeur de la dérivée $y'(x_n)$ en x_n . On trace la tangente à la courbe $y(x)$ en x_n , c'est-à-dire la droite passant par le point (x_n, y_n) et de pente $y'(x_n) = f(x_n, y_n)$. On considère que la tangente approche bien la courbe $y(x)$ sur un petit intervalle $[x_n, x_n + h]$, donc la valeur de $y(x)$ sur cet intervalle est approximativement donnée par la tangente, en particulier :

$$y(x_n + h) \simeq y_n + h y'(x_n).$$



Méthode d'Euler

La méthode d'Euler itère ce processus, autrement dit, on obtient une formule de récurrence, initialisée par x_0 et y_0 , puis :

$$\begin{cases} x_{n+1} = x_n + h \\ y_{n+1} = y_n + hf(x_n, y_n) \end{cases}$$

2.2. Runge-Kutta d'ordre 2

Pour la méthode de Runge-Kutta d'ordre 2 (RK2), chaque pas se fait en deux étapes préalables, puis un calcul de moyenne :

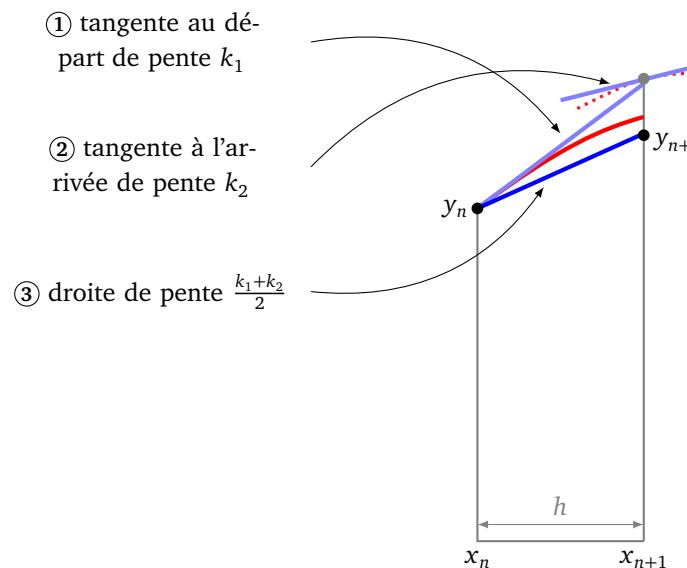
1. On calcule d'abord la pente de la tangente au point de départ $k_1 = f(x_n, y_n)$. On en déduit une approximation de la valeur de y en $x_n + h$, c'est-à-dire $y_n + hk_1$.
2. On calcule ensuite la pente de la tangente au point d'arrivée : $k_2 = f(x_n + h, y_n + hk_1)$.
3. On prend pour valeur de y en $x_n + h$ la valeur correspondant à la moyenne des ces deux pentes : $y_{n+1} = y_n + \frac{h}{2}(k_1 + k_2)$.

La méthode de Runge-Kutta d'ordre 2 itère ce processus. La formule combinant ces étapes est :

$$y(x_n + h) \simeq y_n + \frac{h}{2} \left(f(x_n, y_n) + f(x_n + h, y_n + hk_1) \right).$$

La formule de récurrence est :

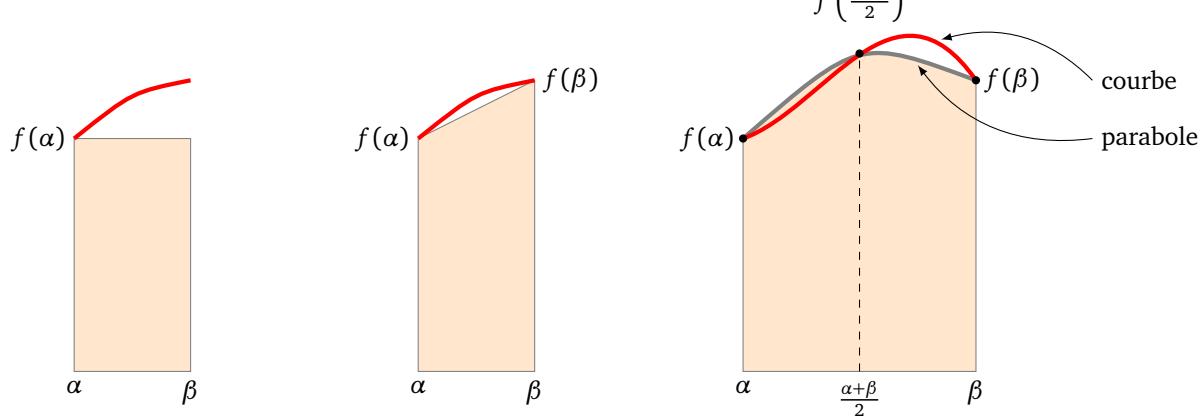
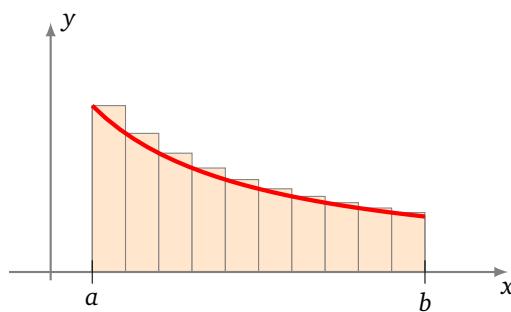
$$\begin{cases} k_1 = f(x_n, y_n) \\ k_2 = f(x_n + h, y_n + hk_1) \\ x_{n+1} = x_n + h \\ y_{n+1} = y_n + \frac{h}{2}(k_1 + k_2) \end{cases}$$



Méthode de Runge-Kutta d'ordre 2

Pourquoi cette méthode est-elle meilleure que la méthode d'Euler ? La méthode d'Euler est basée sur la tangente au point initial et fait partir l'approximation franchement au-dessus ou en-dessous du graphe de la solution. Par contre avec la méthode de Runge-Kutta d'ordre 2, la direction suivie combine la pente au départ et à l'arrivée et est beaucoup plus proche de la solution exacte. On fera une comparaison numérique entre les méthodes dans une section suivante. Il existe une variante de la méthode de Runge-Kutta d'ordre 2 où la pente utilisée est la pente calculée en $x_n + \frac{h}{2}$.

Si on fait une analogie avec le calcul approché d'intégrale : la méthode d'Euler s'apparente à la méthode des rectangles, la méthode RK2 avec la méthode des trapèzes, la méthode RK4 avec la méthode de Simpson.



Méthode des rectangles

Méthode des trapèzes

Méthode de Simpson

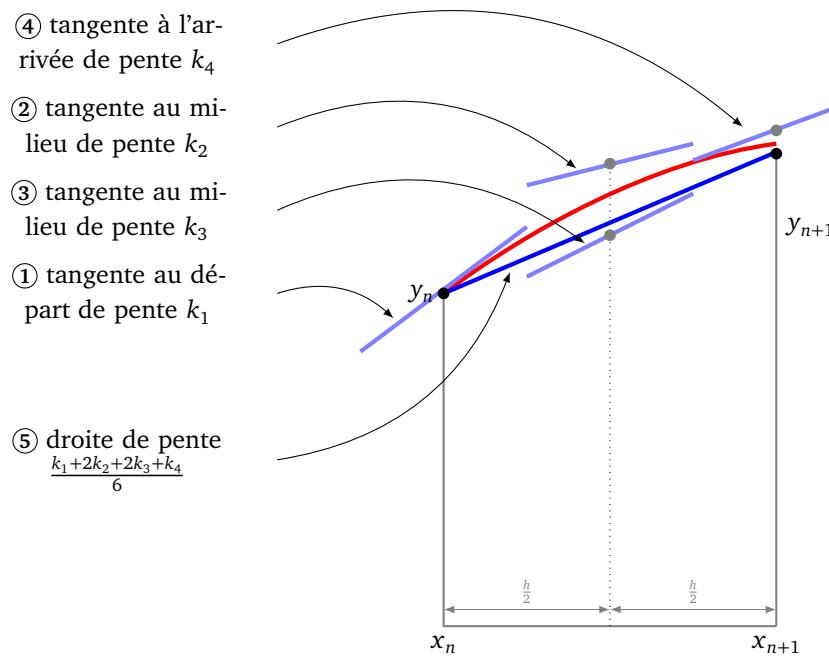
2.3. Runge-Kutta d'ordre 4

La méthode de Runge-Kutta d'ordre 4 (RK4) est encore plus précise que la méthode d'ordre 2. C'est l'une des méthodes les plus populaires. La formule est obtenue à l'aide d'une moyenne pondérée de quatre pentes de tangentes. Voici sa formule de récurrence $x_{n+1} = x_n + h$ et :

$$y_{n+1} = y_n + \frac{h}{6} (k_1 + 2k_2 + 2k_3 + k_4)$$

avec :

$$\begin{cases} k_1 &= f(x_n, y_n) \\ k_2 &= f(x_n + \frac{h}{2}, y_n + \frac{h}{2}k_1) \\ k_3 &= f(x_n + \frac{h}{2}, y_n + \frac{h}{2}k_2) \\ k_4 &= f(x_n + h, y_n + hk_3) \end{cases}$$



Méthode de Runge-Kutta d'ordre 4

2.4. Comparaisons

À chaque étape nos méthodes font des approximations. Plus le pas h est petit, plus l'erreur commise est petite. Ces erreurs se cumulent lors des itérations. On peut donc comparer les méthodes en fonction de l'erreur commise à chaque étape et de l'erreur totale commise à la fin de l'itération. Voici un tableau des ordres de grandeurs des ces erreurs :

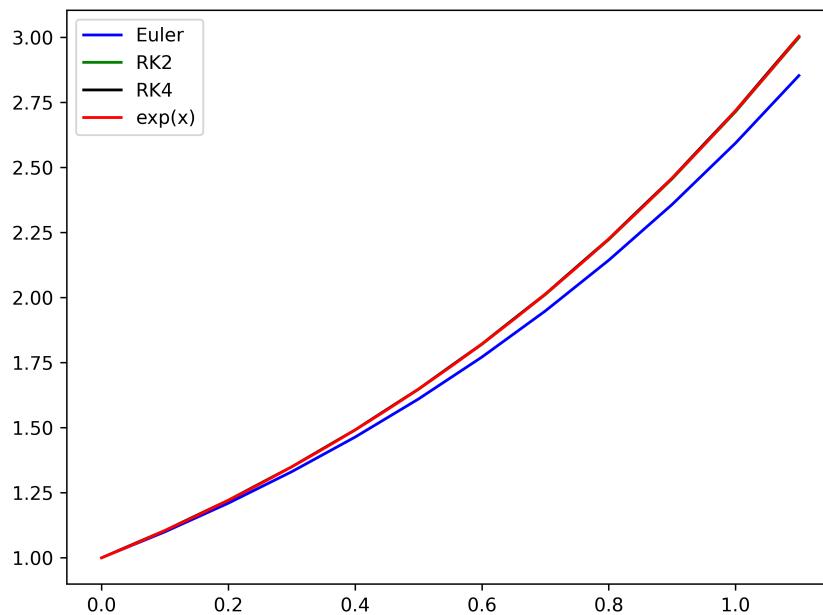
Méthode	erreur locale	erreur globale
Euler	$O(h^2)$	$O(h)$
RK2	$O(h^3)$	$O(h^2)$
RK4	$O(h^5)$	$O(h^4)$

Ainsi, si $h = 0.1$ l'erreur totale commise par la méthode d'Euler est de l'ordre de 0.1, celle de la méthode RK2 est de l'ordre de 0.01 et celle de la méthode RK4 est de l'ordre de 0.0001. On voit donc que les méthodes RK2 et RK4 sont beaucoup plus précises que la méthode d'Euler. Il faut cependant noter que la méthode RK2 évalue la fonction f deux fois à chaque étape et est donc deux fois plus lente que la méthode d'Euler, mais comme l'erreur est quadratique en h , le gain de précision est substantiel.

Voici les valeurs numériques pour le calcul de $\sqrt{2}$ obtenu comme la valeur en 2 de la solution de l'équation différentielle $y' = \frac{1}{2y}$ avec $y(1) = 1$. En gras sont indiquées les décimales exactes.

pas	Euler	RK2	RK4
$h = 0.1$	1.420519799291044	1.4142157109943851	1.414213576569407
$h = 0.01$	1.4148279673659128	1.4142135644527205	1.414213562374446

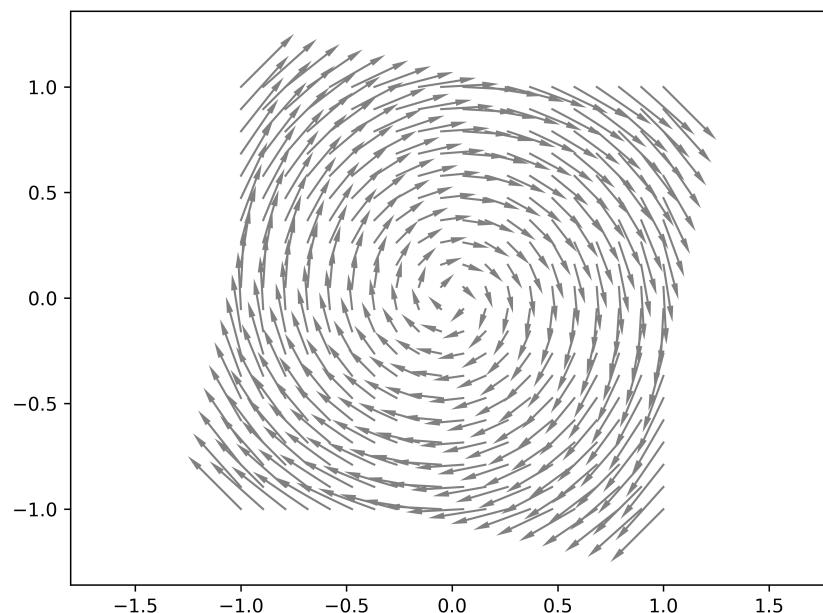
Sur le graphe ci-dessous on dessine les solutions exactes et approchées du problème $y'(x) = y(x)$ avec $y(0) = 1$ pour un pas $h = 0.1$. Les méthodes de Runge-Kutta sont si efficaces que l'on n'arrive pas à distinguer le graphe de la solution exacte ($y(x) = e^x$) avec les solutions approchées RK2 et RK4.



3. Systèmes d'équations différentielles

3.1. Intégration des champs de vecteurs

Un **champ de vecteurs** est la donnée pour chaque point $(x, y) \in \mathbb{R}^2$ du plan d'un vecteur $\vec{F}(x, y)$. Ci-dessous le champ de vecteurs $\vec{F}(x, y) = (y, -x)$.



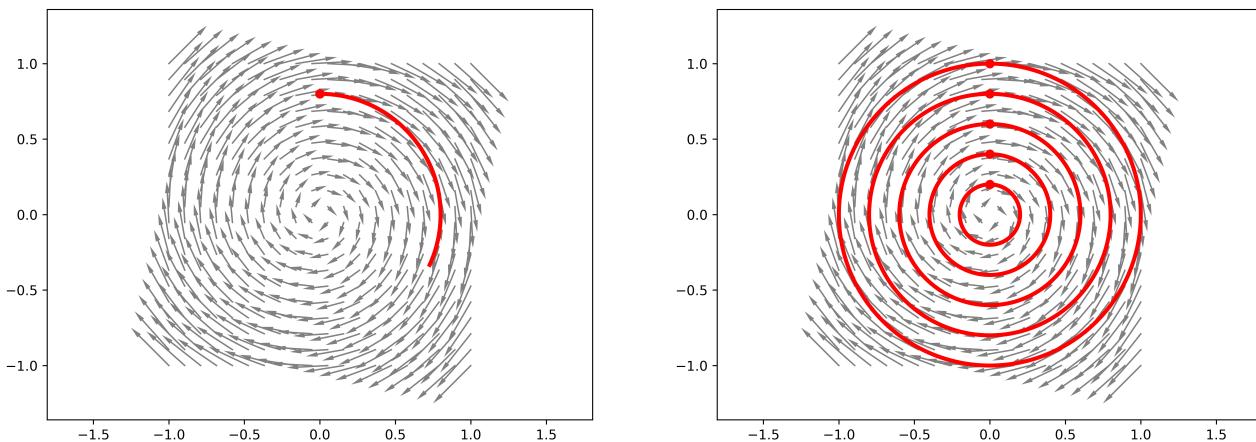
Une **courbe intégrale** est un arc $\gamma(t) = (x(t), y(t))$ tel que pour tout $t \in \mathbb{R}$:

$$(x'(t), y'(t)) = \vec{F}(x, y).$$

Si on note $\vec{F}(x, y) = (f(x, y), g(x, y))$ alors cela revient à trouver une solution du système différentiel suivant :

$$\begin{cases} x'(t) &= f(x(t), y(t)) \\ y'(t) &= g(x(t), y(t)) \end{cases}$$

Graphiquement, en partant d'une position initiale (x_0, y_0) , la courbe intégrale correspond à suivre les flèches du champ de vecteurs. Ci-dessous : une portion d'une courbe intégrale (à gauche), les courbes intégrales sont en fait des paramétrisations de cercles (à droite).



La méthode d'Euler s'étend sans difficulté à ce cas.

Méthode d'Euler.

Soit $x_0, y_0 \in \mathbb{R}$ et $h > 0$.

$$\begin{cases} x_{n+1} &= x_n + hf(x_n, y_n) \\ y_{n+1} &= y_n + hg(x_n, y_n) \end{cases}$$

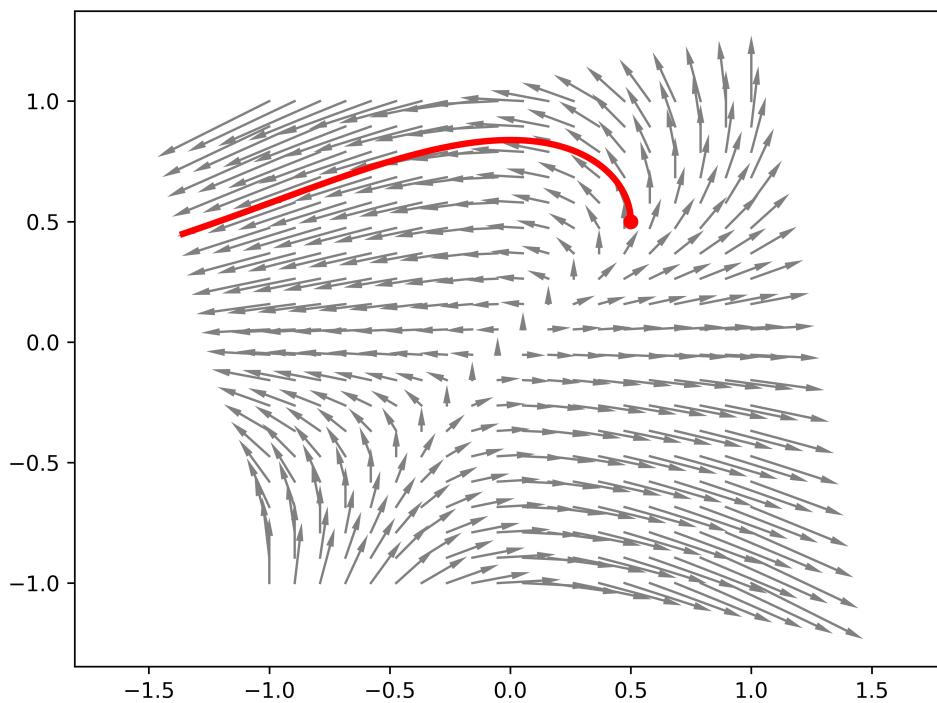
Voici la méthode de Runge-Kutta d'ordre 2.

Méthode RK2.

Soit $x_0, y_0 \in \mathbb{R}$ et $h > 0$.

$$\begin{cases} k_1 &= f(x_n, y_n) \\ \ell_1 &= g(x_n, y_n) \\ k_2 &= f(x_n + hk_1, y_n + h\ell_1) \\ \ell_2 &= g(x_n + hk_1, y_n + h\ell_1) \\ x_{n+1} &= x_n + \frac{h}{2}(k_1 + k_2) \\ y_{n+1} &= y_n + \frac{h}{2}(\ell_1 + \ell_2) \end{cases}$$

Voici un autre exemple de champ de vecteurs : $\vec{F}(x, y) = (x - y, xy)$.



3.2. Équations différentielles d'ordre 2

Considérons une équation différentielle d'ordre 2, c'est-à-dire faisant intervenir $y(x)$, $y'(x)$ et $y''(x)$:

$$y''(x) = f(x, y, y').$$

Elle se transforme en un système de deux équations différentielles d'ordre 1. On pose $z(x) = y'(x)$ et on a alors :

$$\begin{cases} y'(x) &= z(x) \\ z'(x) &= f(x, y, z) \end{cases}$$

La solution est généralement unique une fois que l'on a fixé les conditions initiales $y(x_0)$ et $y'(x_0)$. On peut alors appliquer les méthodes numériques précédentes.

Le principe fondamental de la mécanique conduit naturellement à un système d'équations différentielles d'ordre 2. En effet la vitesse est la dérivée de la position et l'accélération est la dérivée seconde de la position. Appliquons ceci au mouvement d'un satellite orbitant dans le plan orbital de la Terre et du Soleil (supposés tous les deux fixes!). Le satellite est soumis à la force d'attraction de la Terre et à celle du Soleil. Ces forces s'écrivent :

$$\vec{F}_T = -\frac{GmM_T}{r_T^2} \vec{e}_T \quad \vec{F}_S = -\frac{GmM_S}{r_S^2} \vec{e}_S$$

et le principe fondamental de la mécanique donne :

$$\vec{F}_T + \vec{F}_S = m \vec{\alpha}$$

Les données sont :

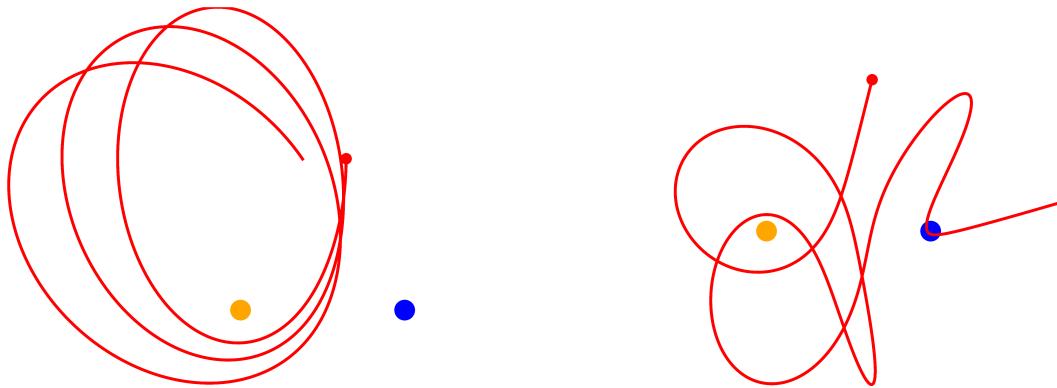
- m : masse du satellite,
- M_T : masse de la Terre,
- M_S : masse du Soleil,
- G : constante gravitationnelle,
- r_T : distance du satellite à la Terre,
- r_S : distance du satellite au Soleil,
- \vec{e}_T : vecteur unitaire issu de la Terre dans la direction du satellite,
- \vec{e}_S : vecteur unitaire issu du Soleil dans la direction du satellite.

Si $(x(t), y(t))$ est la position du satellite, alors, le principe fondamental, projeté sur l'axe x , donne :

$$x''(t) = -\frac{GM_T(x(t)-x_S)}{\left((x(t)-x_S)^2 + (y(t)-y_S)^2\right)^{\frac{3}{2}}} - \frac{GM_S(x(t)-x_S)}{\left((x(t)-x_S)^2 + (y(t)-y_S)^2\right)^{\frac{3}{2}}}$$

On obtient une équation similaire pour $y''(t)$. On transforme ces deux équations d'ordre 2 en un système de 4 équations d'ordre 1.

Voici des exemples d'orbites que l'on peut obtenir.



Fractales

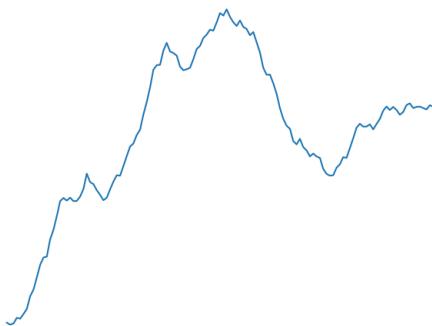
Les fractales sont des formes géométriques auto-similaires : lorsque l'on zoomé sur une partie, on retrouve une image ressemblant à la figure globale. Les structures fractales permettent de dessiner des paysages et de la végétation. La méthode est facile à implémenter, permet de générer aléatoirement une grande variété de structures, utilise très peu de données, mais par contre nécessite des calculs.

Dans ce chapitre on se contente d'une modélisation assez grossière : sans couleur ni texture.

1. Montagnes

Dessinons un paysage avec des montagnes et des vallées. La génération est rapide, aléatoire et paramétrable. Afin de mieux comprendre l'algorithme général, on commence avec une seule dimension par la génération du profil d'une montagne, ou bien d'un cours de bourse.

1.1. Une dimension

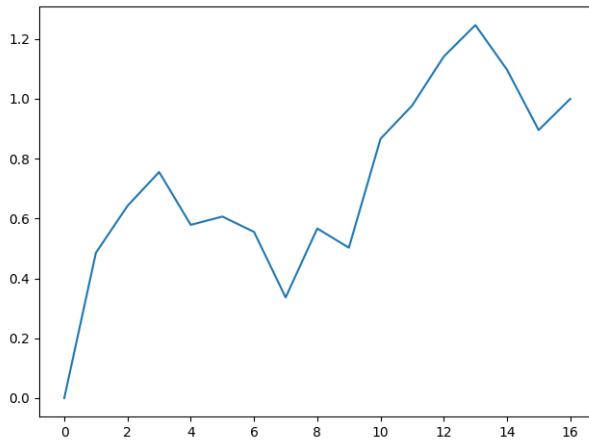


Données. Voici les données qui permettent de générer un profil de montagne :

- une altitude de départ H_a ,
- une altitude d'arrivée H_b ,
- une valeur d'amplitude maximale h_0 ,
- un coefficient de rugosité r ,
- un nombre de subdivisions $N = 2^n$.

On peut aussi fixer le germe (*seed*) du processus de génération pseudo-aléatoire afin de pouvoir reproduire les mêmes résultats lors d'une génération future.

Sortie. L'algorithme renvoie $N + 1$ valeurs (la première étant H_a et la dernière H_b).



Un exemple avec $n = 5$, $N = 16$, $H_a = 0$, $H_b = 1$, $h_0 = 1$ et $r = 0.7$.

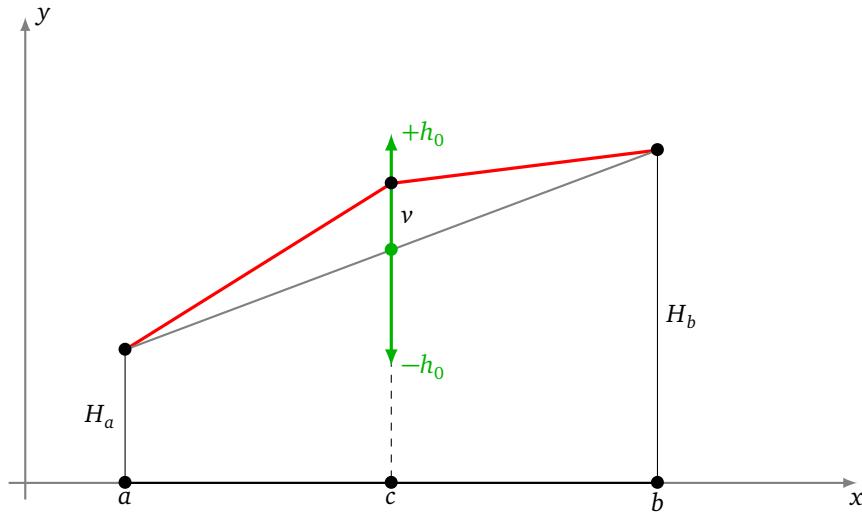
Algorithme.

On considère qu'on trace le profil au-dessus d'un segment $[a, b]$. L'algorithme est un calcul de moyenne des altitudes avec ajout d'un saut aléatoire. Il s'effectue par dichotomies successives de l'intervalle de départ.

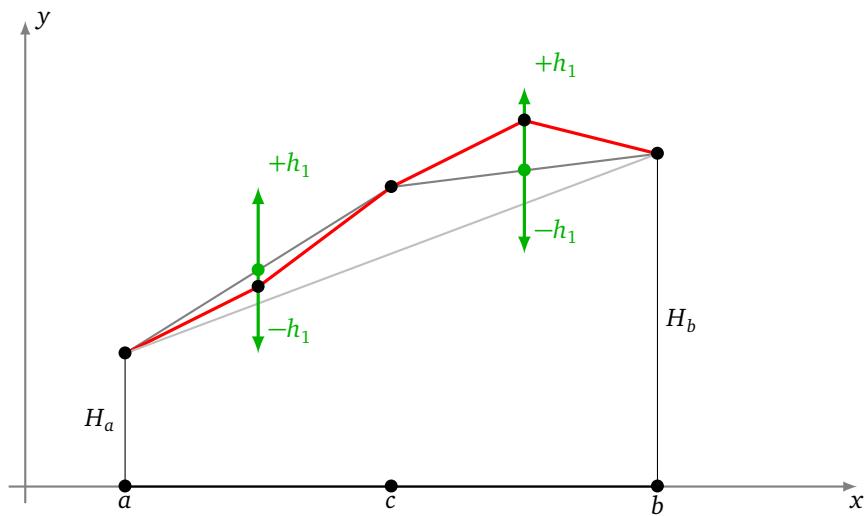
- *Étape 1.* En $c = \frac{a+b}{2}$, qui est le milieu de $[a, b]$, on calcule la moyenne des hauteurs de départ et d'arrivée à laquelle on ajoute une valeur aléatoire :

$$H_c = \frac{H_a + H_b}{2} + v.$$

La valeur v est tirée aléatoirement dans l'intervalle $[-h_0, +h_0]$. On obtient donc 3 ($= 2^1 + 1$) valeurs H_a, H_c, H_b .



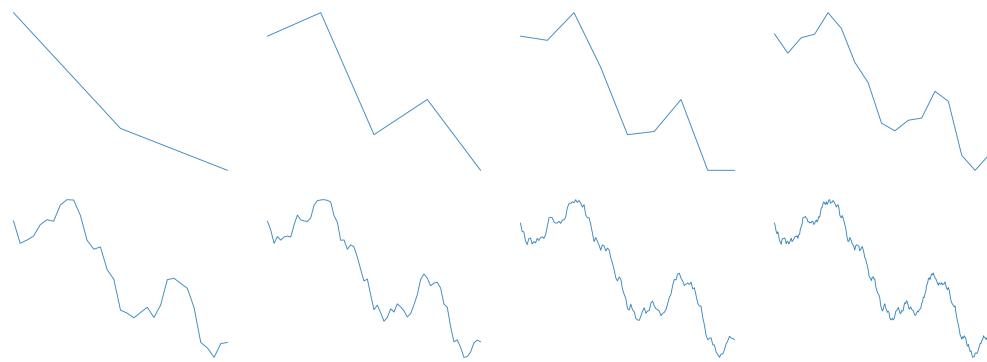
- *Étape 2.* On recommence entre a et c en calculant la moyenne entre H_a et H_c à laquelle on ajoute une valeur aléatoire de $[-h_1, +h_1]$ où $h_1 = \frac{h_0}{2^r}$. On fait de même entre c et b , avec la moyenne entre H_c et H_b et une nouvelle valeur aléatoire de $[-h_1, +h_1]$. On obtient 5 ($= 2^2 + 1$) altitudes.



- ...
- Étape i . On obtient $2^i + 1$ altitudes. La hauteur aléatoire est choisie dans $[-h_i, +h_i]$ où $h_i = \frac{h_0}{2^{ir}}$.
- Étape n . On obtient $2^n + 1$ altitudes.

Exemple.

Voici le déroulé de la construction étape par étape.

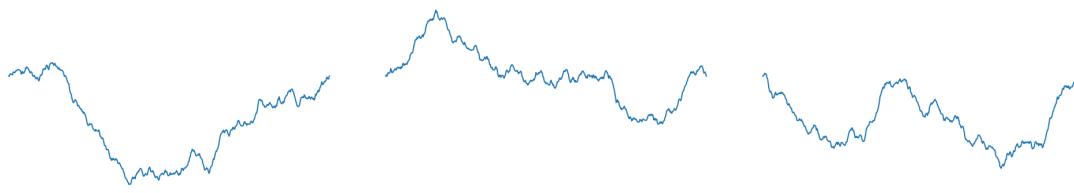


Les étapes de la construction avec n variant de 1 à 8.

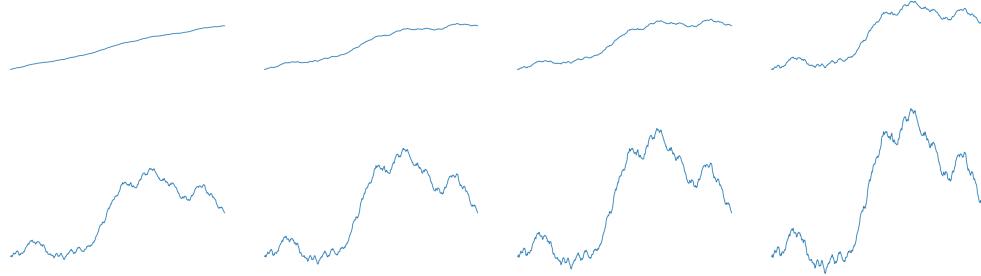
Exemple.

Voici les profils obtenus pour différents tirages aléatoires. Les valeurs H_a , H_b , h_0 et r sont les mêmes pour tous ces profils.



**Exemple.**

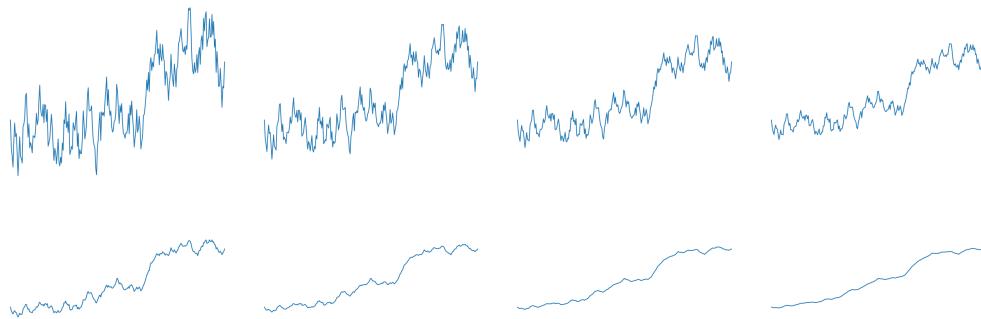
Voici les profils obtenus pour différentes valeurs de l'amplitude h_0 . Les autres valeurs (y compris le germe du tirage pseudo-aléatoire) sont les mêmes.



L'amplitude de variation aléatoire h_0 varie : 0.1, 0.3, 0.5, 1.0, 1.5, 2.0, 2.5, 3.0.

Exemple.

Voici les profils obtenus pour différentes valeurs de la rugosité r . Les autres valeurs (y compris le germe du tirage pseudo-aléatoire) sont les mêmes. Plus le coefficient r est grand, plus la courbe est lisse.



Le coefficient de rugosité r varie : 0.2, 0.3, 0.4, 0.5, 0.7, 0.9, 1.1, 1.3.

Exercice.

Programmer cet algorithme sous la forme itérative comme expliqué ci-dessus. Programmer aussi une version récursive.

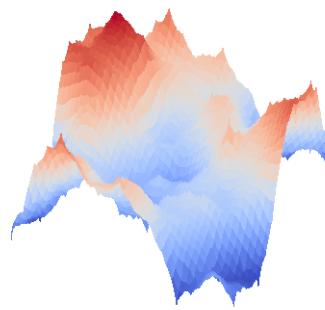
1.2. Deux dimensions

Tracer une surface dans l'espace qui représente un paysage est évidemment la situation qui nous intéresse le plus. Il s'agit d'adapter l'algorithme précédent avec une dimension supplémentaire afin de générer les altitudes d'un terrain.

Données. Voici les données qui permettent de générer un profil de montagne au-dessus d'une zone carrée :

- une altitude de départ H qui sera l'altitude aux quatre coins du carré,
- une valeur d'amplitude maximale h_0 ,
- un coefficient de rugosité r ,
- un nombre de subdivisions $N = 2^n$.

Sortie. L'algorithme renvoie $(N + 1)^2$ valeurs, correspondant aux altitudes au-dessus d'une grille carrée $(N + 1) \times (N + 1)$.



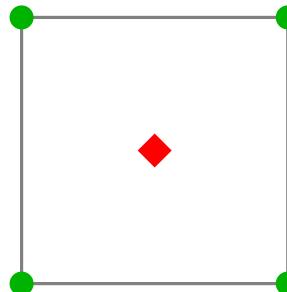
Algorithme.

La construction se fait en itérant une construction « diamant-carré ».

- *Étape diamant.* À partir de 4 valeurs connues aux sommets d'un carré, on détermine une valeur au centre selon la formule :

$$\text{moyenne des 4 valeurs} + \text{valeur aléatoire.}$$

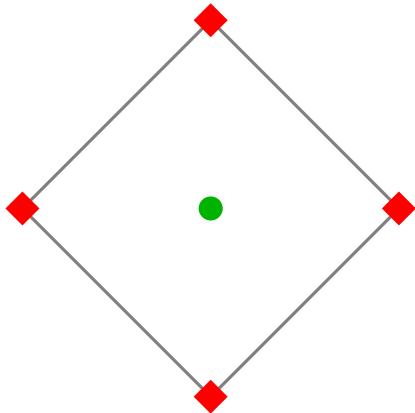
À l'étape numéro i la valeur aléatoire est dans l'intervalle $[-h_i, +h_i]$ où $h_i = \frac{h_0}{2^{ir}}$.



- *Étape carré.* À partir de 4 valeurs connues aux sommets d'un losange, on détermine une valeur au centre selon la formule :

$$\text{moyenne des 4 valeurs} + \text{valeur aléatoire.}$$

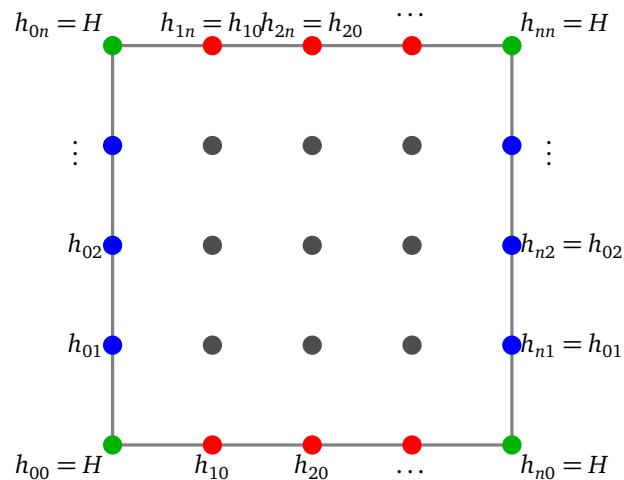
À l'étape numéro i la valeur aléatoire est de nouveau choisie dans l'intervalle $[-h_i, +h_i]$.



L'étape carré est en fait une étape diamant composée avec une rotation de 45° .

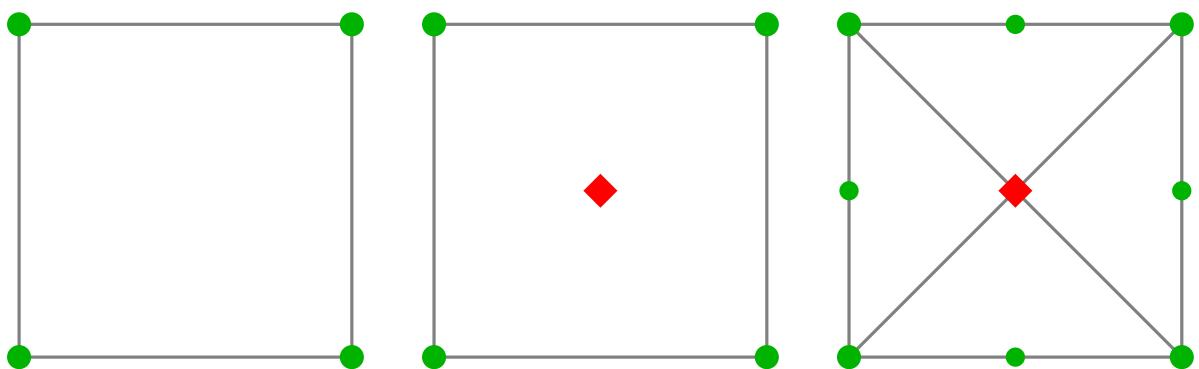
Il y a un problème avec l'étape carré car, aux bords de la zone carrée initiale, on ne dispose pas d'un losange complet et donc on n'a pas toujours 4 valeurs à moyenner. Plusieurs solutions sont possibles :

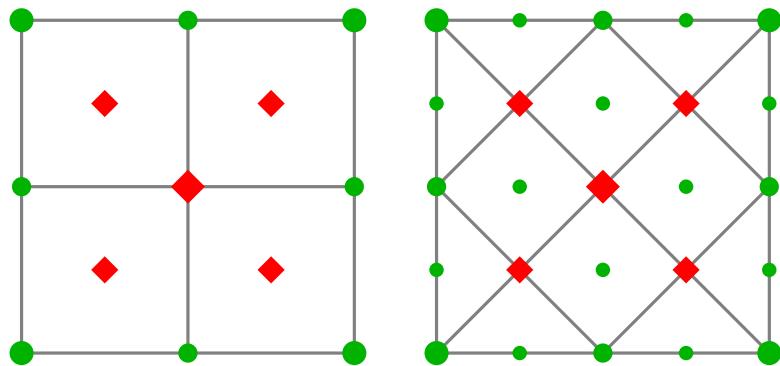
- ne tenir compte que des sommets présents en faisant la moyenne sur 3 sommets au lieu de 4 ;
- considérer que les valeurs sont périodiques : les valeurs du côté gauche du tableau initial sont les mêmes que les valeurs du côté droit, et les valeurs du bas sont les mêmes que les valeurs du haut.



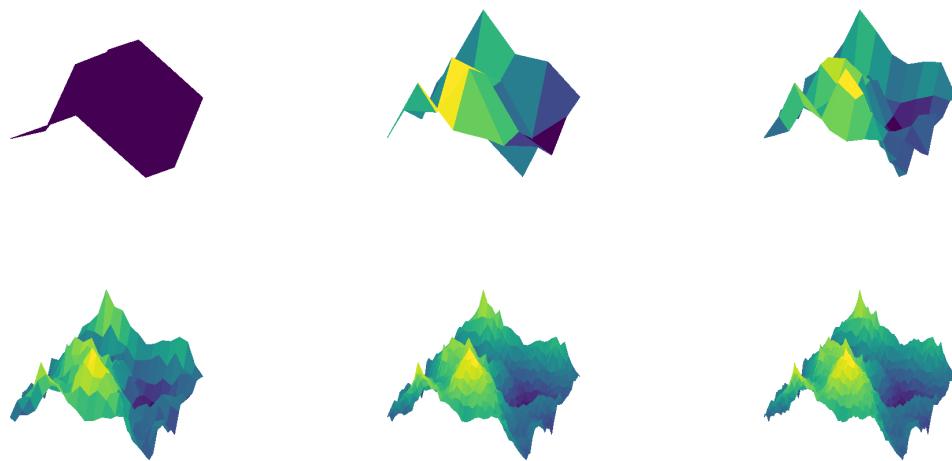
Nous adoptons la seconde solution qui a l'avantage de créer une surface avec le bord gauche identique au bord droit et le bord haut identique à celui du bas et donne ainsi un motif de base qui se recolle parfaitement lorsque qu'on en juxtapose plusieurs.

Voici les points où les altitudes sont calculées lors des premières étapes de la construction « diamant-carré ».



**Exemple.**

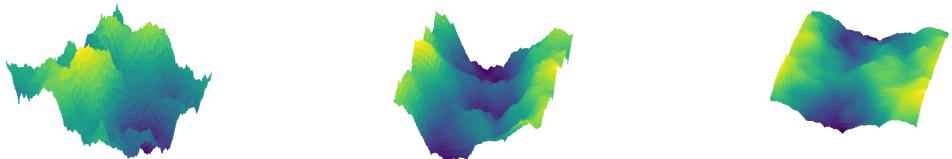
Voici un exemple de cette construction étape par étape.



Les étapes de la construction avec n variant de 1 à 6.

Exemple.

Voici les profils obtenus pour différents tirages aléatoires et différentes valeurs des paramètres.

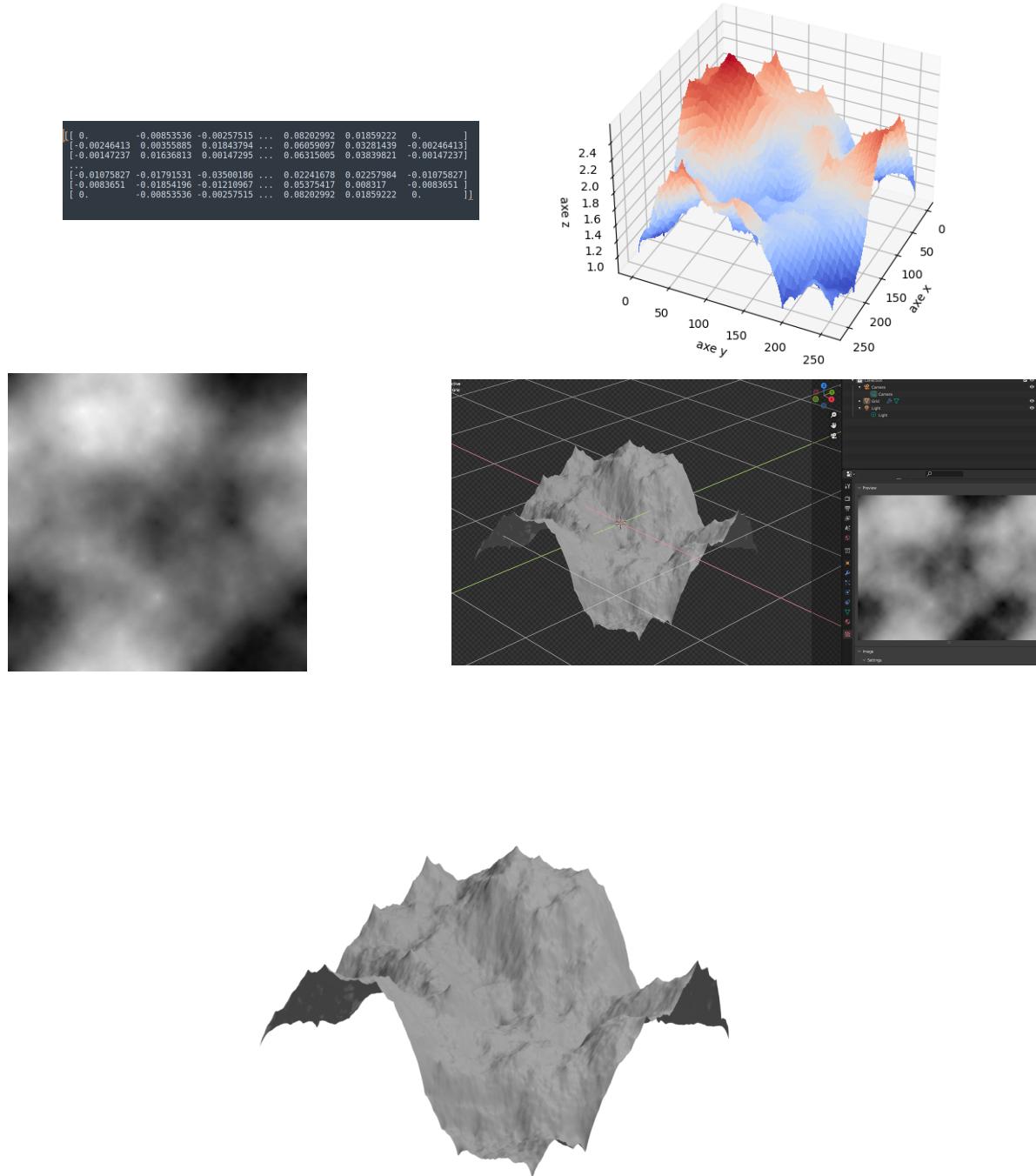
**Carte de hauteurs – *heightmap*.**

Comment utiliser les surfaces obtenues ? Tout d'abord les données produites se résument à un tableau de taille $(N + 1) \times (N + 1)$ où chaque entrée en (i, j) est un nombre réel représentant l'altitude du terrain au-dessus du point (i, j) . Les données se transforment aisément en une image en niveaux de gris, appelée **carte de hauteurs (*heightmap*)**, sa taille est $(N + 1) \times (N + 1)$ et chaque niveau de gris représente une hauteur mise à l'échelle et arrondie. Par exemple 0 (noir) pour l'altitude la plus basse et 255 (blanc) pour l'altitude la plus haute.

Cette image peut ensuite être importée dans un logiciel de traitement d'images (par exemple *Blender*) pour reconstituer le paysage et obtenir un rendu 3D avec éclairage, texture...

Exemple.

Construction d'une carte de hauteurs et rendu 3D.



Niveau de la mer.

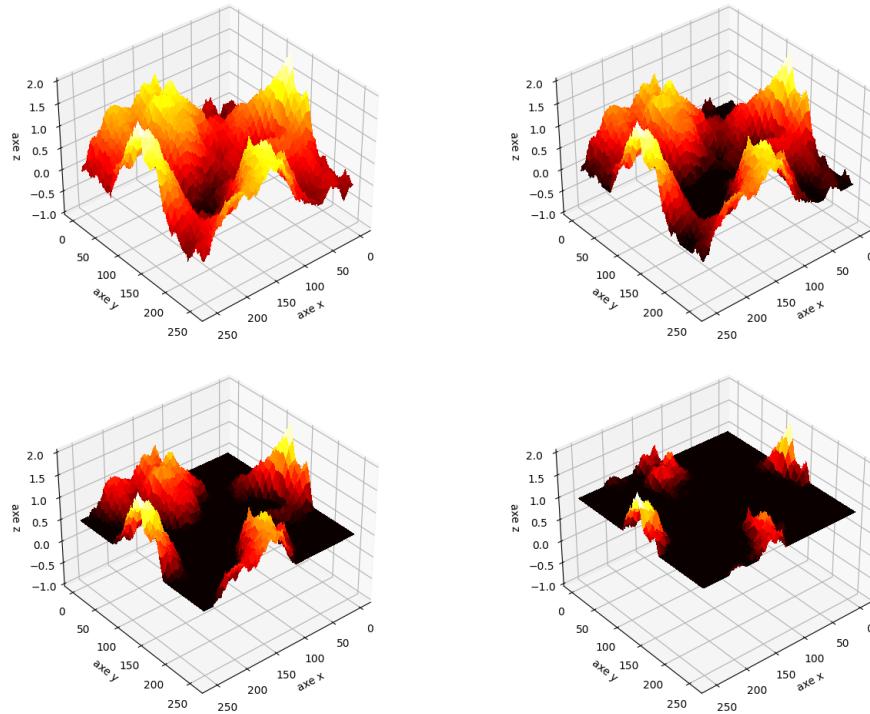
On peut créer un paysage de bord de mer : tout ce qui a une hauteur négative est ramené à l'altitude 0 par la formule $h' = \max(h, 0)$. Dans la pratique on fixe le niveau de l'eau à une altitude h_e et on applique la formule :

$$h' = \max(h, h_e)$$

qui conserve la hauteur h si on est au-dessus de la mer et ramène l'altitude à h_e sinon.

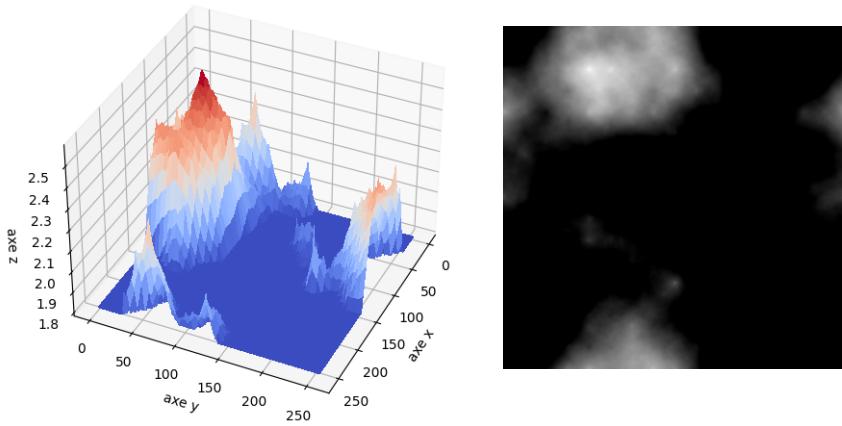
Exemple.

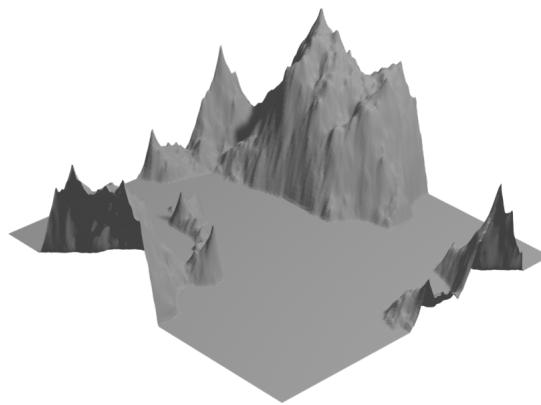
Voici le changement de paysage lorsque le niveau de l'eau monte.



Exemple.

Avec rendu 3D



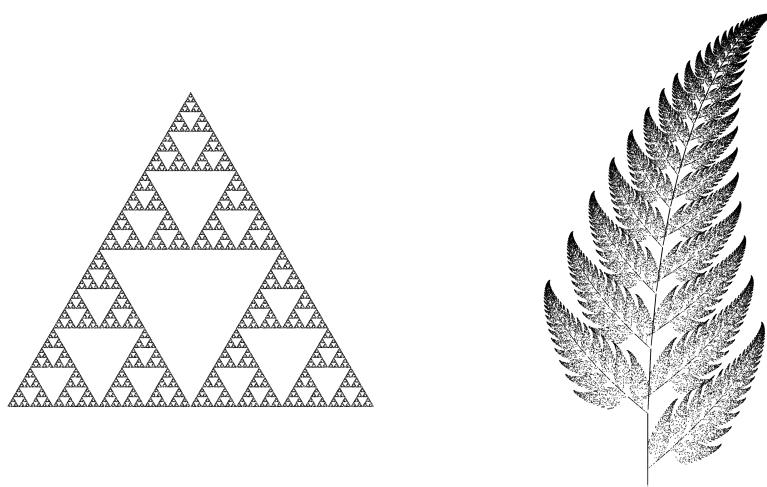


(a) La visualisation Python avec matplotlib. (b) L'image en niveaux de gris (heightmap). (c) Rendu 3D.

2. IFS

2.1. La fougère de Barnsley

Les IFS (pour *Iterated Functions System*) sont des fractales que l'on trace point par point. Elles permettent de dessiner très facilement des figures géométriques comme le triangle de Sierpinski, mais aussi des objets de la nature, en particulier des feuilles comme la fougère de Barnsley.



Ce qui est surprenant, c'est que ces dessins sont obtenus par un processus aléatoire et nécessitent très peu de données, par exemple la fougère est dessinée à partir de 24 nombres réels seulement.

2.2. Transformations du plan

C'est le moment d'aller relire la section « Transformations du plan » du chapitre « Matrices ». On rappelle juste ici la définition générale.

Une **transformation affine** du plan est l'application $F : \mathbb{R}^2 \rightarrow \mathbb{R}^2$ définie par :

$$\begin{pmatrix} x \\ y \end{pmatrix} \mapsto \begin{pmatrix} a & b \\ c & d \end{pmatrix} \begin{pmatrix} x \\ y \end{pmatrix} + \begin{pmatrix} e \\ f \end{pmatrix},$$

où a, b, c, d, e, f sont des réels quelconques.

En d'autres termes, l'image d'un point (x, y) du plan est le point $F(x, y) = (x', y')$ avec

$$\begin{cases} x' = ax + by + e \\ y' = cx + dy + f \end{cases}.$$

En fait, une transformation affine F est la composée d'une transformation linéaire

$$\begin{pmatrix} x \\ y \end{pmatrix} \mapsto A \begin{pmatrix} x \\ y \end{pmatrix} \quad \text{où } A = \begin{pmatrix} a & b \\ c & d \end{pmatrix}$$

suivie d'une translation

$$\begin{pmatrix} x \\ y \end{pmatrix} \mapsto \begin{pmatrix} x \\ y \end{pmatrix} + B \quad \text{où } B = \begin{pmatrix} e \\ f \end{pmatrix}.$$

2.3. Algorithme

La fractale associée à un système itéré de fonctions s'appelle un **attracteur**. Comment tracer l'attracteur d'un système itéré de fonctions ? Soient F_1, F_2, \dots, F_ℓ des transformations affines. La méthode la plus efficace afin de tracer la fractale est appelée « jeu du chaos » : partant d'un point quelconque $P_0 \in \mathbb{R}^2$, on construit une suite de points par récurrence. Si P_k est construit, alors on choisit aléatoirement l'une des transformations F_i et on définit $P_{k+1} = F_i(P_k)$. On laisse de côté les premiers points (disons les 100 premiers qui peuvent être « loin » de l'attracteur) et on trace les points suivants.

Algorithme.

Algorithme « jeu du chaos ».

Entrée : une famille $\{F_1, F_2, \dots, F_\ell\}$ de transformations affines.

Sortie : un tracé approché de la fractale.

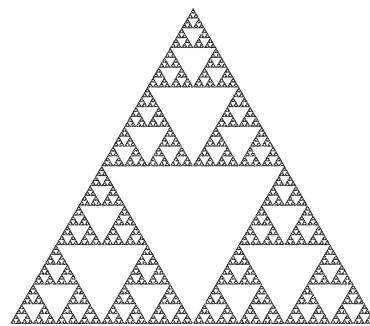
- Fixer $P_0 \in \mathbb{R}^2$ au hasard.
- Pour $k \in \{0, \dots, N_{\max}\}$:
 - on choisit au hasard $i \in \{1, \dots, \ell\}$,
 - on pose $P_{k+1} = F_i(P_k)$,
 - à partir de $k \geq N_{\min}$ on affiche ce point.

N_{\max} représente le nombre de points à calculer (par exemple $N_{\max} = 10\,000$). N_{\min} est une constante (par exemple $N_{\min} = 100$) qui permet de ne pas tracer les tout premiers points qui peuvent être loin de l'attracteur.

Probabilités. En plus, pour optimiser les calculs, on ne choisit pas toujours les transformations F_i de façon équiprobable mais avec une probabilité p_i où $p_1 + p_2 + \dots + p_\ell = 1$.

2.4. Exemples

Le triangle de Sierpinski.



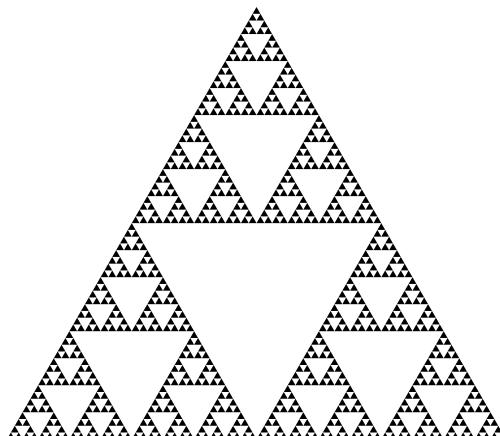
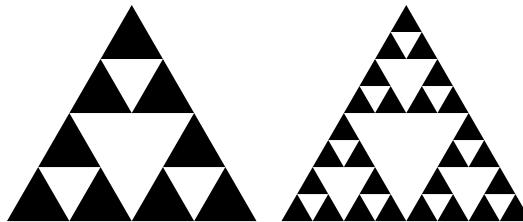
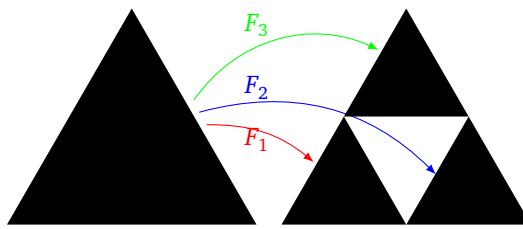
Voici les valeurs numériques (approchées) qui définissent cette fractale.

	a	b	c	d	e	f	p
F_1	0.5	0	0	0.5	0	0	0.33
F_2	0.5	0	0	0.5	0.5	0	0.33
F_3	0.5	0	0	0.5	0.25	0.43	0.33

Pour vraiment comprendre l'action des trois fonctions, partons d'un triangle équilatéral S_0 dont les sommets sont $(0, 0)$, $(1, 0)$, $(\frac{1}{2}, \frac{\sqrt{3}}{2})$. On considère les trois transformations affines suivantes :

$$F_1 \begin{pmatrix} x \\ y \end{pmatrix} = \frac{1}{2} \begin{pmatrix} x \\ y \end{pmatrix} \quad F_2 \begin{pmatrix} x \\ y \end{pmatrix} = \frac{1}{2} \begin{pmatrix} x \\ y \end{pmatrix} + \begin{pmatrix} \frac{1}{2} \\ 0 \end{pmatrix} \quad F_3 \begin{pmatrix} x \\ y \end{pmatrix} = \frac{1}{2} \begin{pmatrix} x \\ y \end{pmatrix} + \begin{pmatrix} \frac{1}{4} \\ \frac{\sqrt{3}}{4} \end{pmatrix}$$

Chacune des transformations F_i est la composition d'une homothétie de rapport $\frac{1}{2}$ et d'une translation. À la première étape, chacune envoie donc le grand triangle équilatéral S_0 sur un triangle plus petit. Géométriquement, c'est comme si on retirait un triangle au centre de chaque plus gros triangle. On a représenté S_0 , S_1 , S_2 , S_3 et S_6 , ce qui donne une bonne idée de l'attracteur, appelé le *triangle de Sierpinski*.



La fougère de Barnsley

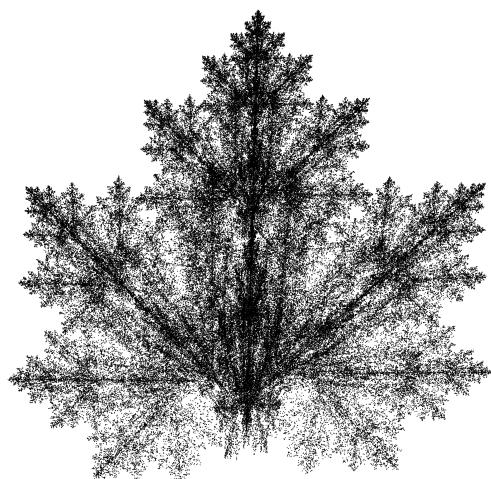


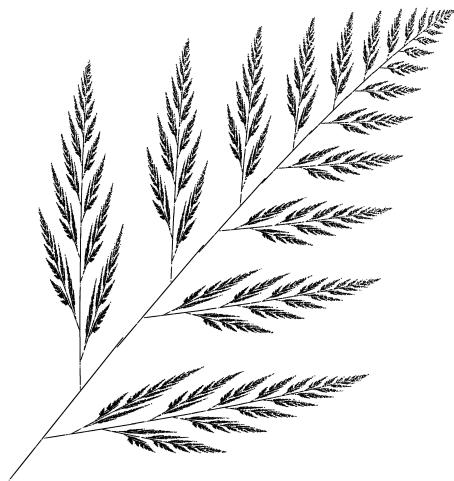
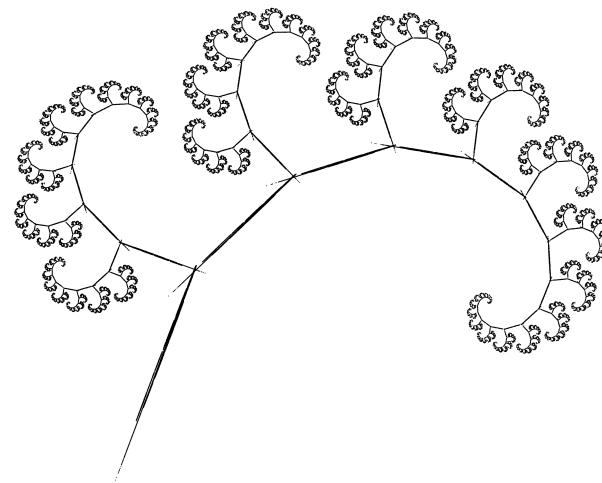
	a	b	c	d	e	f	p
F_1	0	0	0	0.16	0	0	0.01
F_2	0.85	0.04	-0.04	0.85	0	1.6	0.85
F_3	0.2	-0.26	0.23	0.22	0	1.6	0.07
F_4	-0.15	0.28	0.26	0.24	0	0.44	0.07

De part la nature itérative de la construction il est difficile de comprendre ce que fait chaque transformation. Par exemple, la première transformation F_1 est une projection composée avec une homothétie d'un petit rapport. Elle envoie toute la fougère sur la base de la tige (la portion verticale).

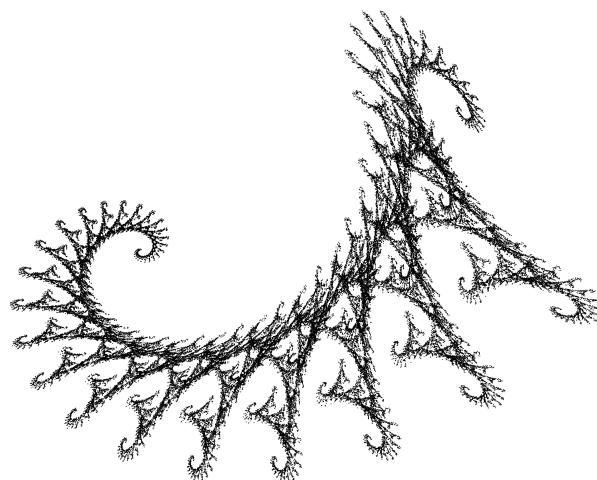
Autres exemples.

D'autres formes de feuilles.

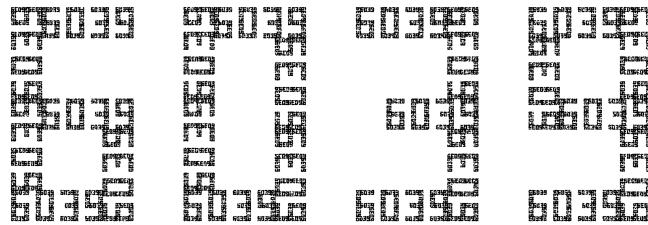




Un hippocampe (?).



Ci-dessous un exemple d'application du « théorème de collage » qui permet d'approcher n'importe quel ensemble voulu comme attracteur d'un système itéré de fonctions.



3. L-systèmes

Les L-systèmes ont été inventés par le botaniste A. Lindenmayer afin de modéliser les plantes. À partir d'un mot initial et d'opérations de remplacement, on arrive à des mots compliqués. Lorsque l'on « dessine » ces mots, on obtient de superbes figures fractales.

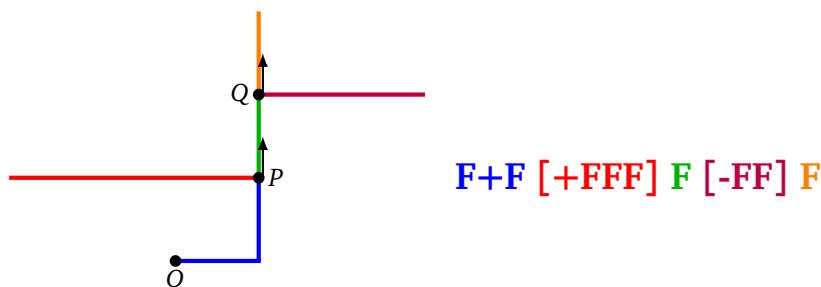
3.1. Tortue

La « tortue » est un langage simple pour dessiner : une tortue se déplace dans le plan (ou l'espace) et trace un trait sur son passage. Elle exécute les instructions selon une suite de lettres :

- **F** ou **G** : avance d'une quantité fixée (en traçant, F pour *forward*),
- **+** : tourne à gauche, sans avancer, d'un angle fixé (le plus souvent 90°),
- **-** : tourne à droite d'un angle fixé.
- **[** : mémorise la position de la tortue (coordonnées (x, y) et direction).
- **]** : repart de la position mémorisée par le crochet [correspondant.
- **X, Y** : ne font rien.

Exemple.

Comprendons l'exemple du tracé du **F+F [+FFF] F [-FF] F**.



- **F+F** : on part du point O , on avance, on tourne, on avance.
- **[+FFF]** : on retient la position actuelle (le point P) et aussi la direction ; on tourne, on avance trois fois (on trace le segment rouge) ; à la fin on replace la tortue à la position P (sans tracer et avec la même direction que celle auparavant).
- **F** : depuis P on avance (segment vert).
- **[-FF]** : on retient la position Q et la direction, on tourne et on trace le segment violet. On revient en Q

avec l'ancienne direction.

- **F** : depuis Q on trace le dernier segment.

Exercice.

Programmer une fonction qui, en entrée reçoit une chaîne de caractères, et trace les instructions correspondantes.

Indications. Avec *Python* il existe une librairie *turtle* qui fournit les instruction `forward()`, `left()`, `right()`.

Pile. Il est beaucoup plus délicat de gérer les crochets ! La méthode qui rend les choses très simples est d'utiliser une « pile » (concept que l'on n'expliquera pas ici).

- Au départ la pile est vide.
- On lit un par un les caractères du mot.
- Si le caractère est le crochet ouvrant « [» alors on ajoute à la pile la position et la direction courante de la tortue $((x, y), \theta)$ (que l'on obtient par `(position(), heading())` du module *turtle*).
- Si le caractère est le crochet fermant «] » alors on dépile (dépiler c'est lire l'élément du haut de la pile et le retirer). On met la position de la tortue et l'angle avec les valeurs lues (utiliser `goto()` et `setheading()`).

3.2. Chercher–Remplacer–Itérer

Un **L-système** est la donnée d'un mot initial et de règles de remplacement. Voici un exemple avec le mot de départ et une seule règle :

initialisation : **G+F-G** règle : **F → FGF**

Le **k-ème itéré** du L-système s'obtient en appliquant k fois la substitution au mot de départ. Avec notre exemple :

- Première itération. Le mot de départ est **G+F-G**, la règle est **F → FGF** : on remplace le **F** par **FGF**. On obtient le mot **G+FGF-G**.
- Deuxième itération. On part du mot obtenu **G+FGF-G**, on remplace les deux **F** par **FGF** : on obtient le mot **G+FGFGFGF-G**.
- Le troisième itéré est **G+FGFGFGFGFGFGF-G**, etc.

Lorsqu'il y a deux règles (ou plus) il faut les appliquer en même temps. Voici un exemple de L-système à deux règles :

initialisation : **F** règle 1 : **F → G+F** règle 2 : **G → GG**

Avec notre exemple :

- Première itération. Le mot de départ est **F**, on applique la première règle **F → G+F** (la seconde règle ne s'applique pas, car il n'y a pas encore de **G**) : on obtient le mot **G+F**.
- Deuxième itération. On part du mot obtenu **G+F**, on remplace le **F** par **G+F** et en même temps le **G** par **GG** : on obtient le mot **GG+G+F**.
- Le troisième itéré est **GGGG+GG+G+F**, etc.

Exemple.

Considérons le L-système du flocon de Koch :

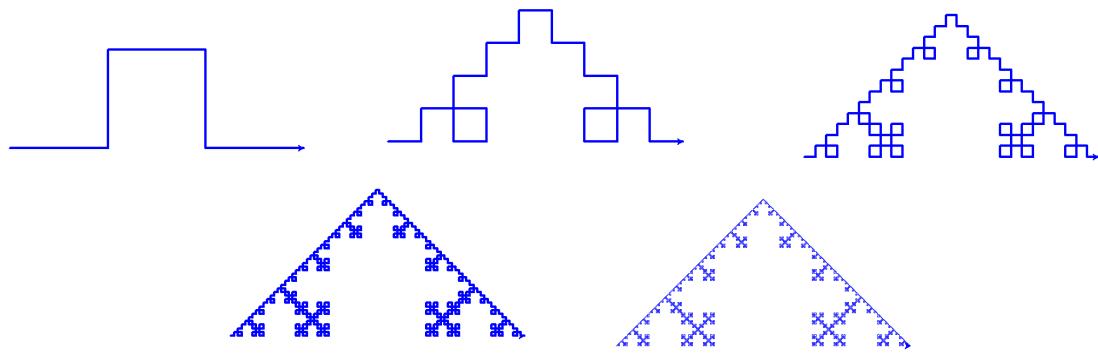
initialisation : **F** règle : **F → F+F-F-F+F**

- le premier itéré ($k = 1$) est **F+F-F-F+F**.
- le deuxième itéré ($k = 2$) est :

F+F-F-F+F+F-F-F+F-F-F+F-F-F+F+F+F+F-F-F+F

- avec le troisième itéré ($k = 3$), on obtient **F+F-F-F+F+...** un mot de 249 lettres.

Voici les images pour $k = 1$ jusqu'à $k = 5$ qui correspondent aux premières étapes d'un flocon de Koch.



Exemple.

Considérons le L-système du triangle de Sierpinski avec deux règles de remplacement :

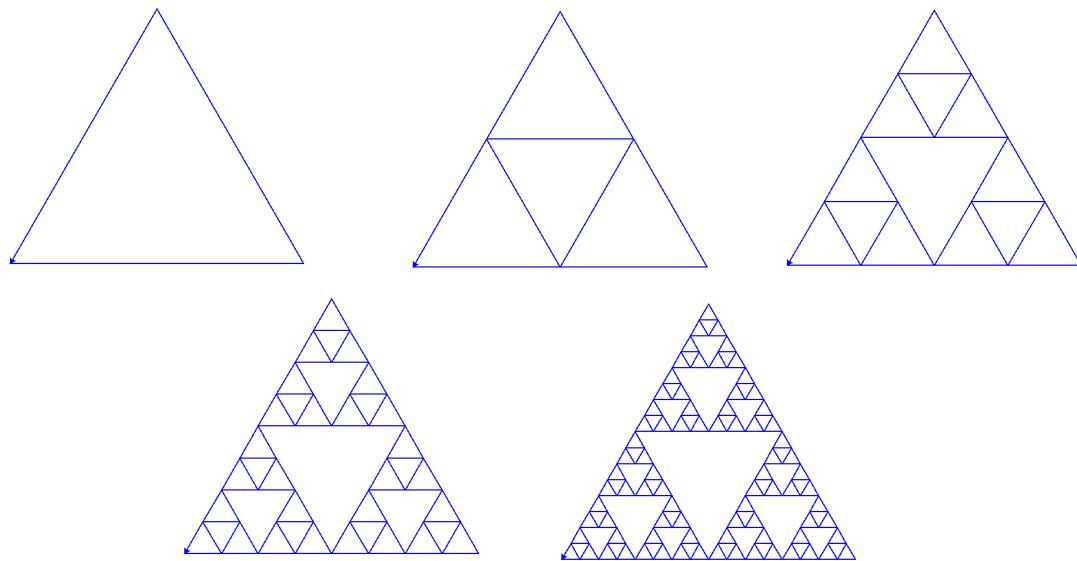
initialisation : **F-G-G** règle 1 : **F** → **F+G+F+G-F** règle 2 : **G** → **GG**

Par exemple, avec :

- $k = 1$, **F-G+G-F-GG-GG**,
- pour $k = 2$:

F-G+G-F-GG+F-G+F+G-F+GG-F-G+F+G-F-GGGG-GGGG

Lorsqu'on trace ces mots (avec un angle de -120° pour l'instruction **+** et 120° pour l'instruction **-**) on obtient la construction du triangle de Sierpinski.

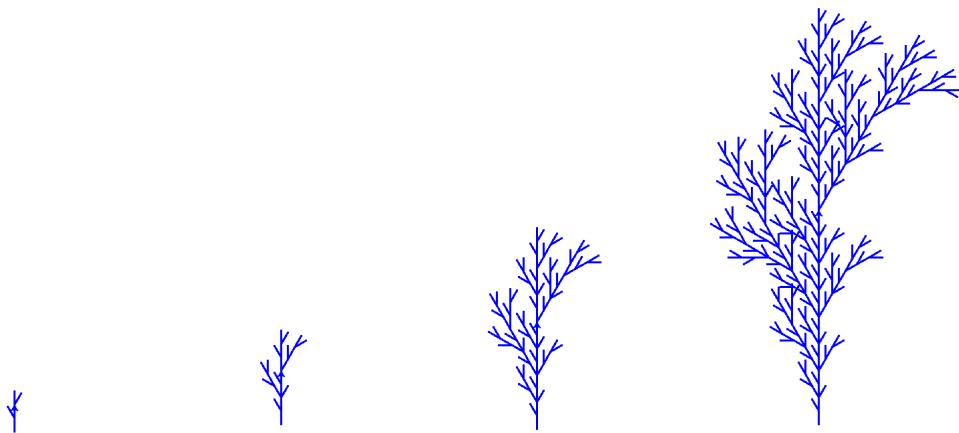


3.3. Plantes en 2D

Voici des exemples de L-systèmes dont le tracé ressemble à des plantes. Pour ces exemples, l'angle est généralement fixé à 30° .

Voici les quatre premiers tracés du L-système défini par :

initialisation : **F** règle : **F** → **F[F+F-F][F]**

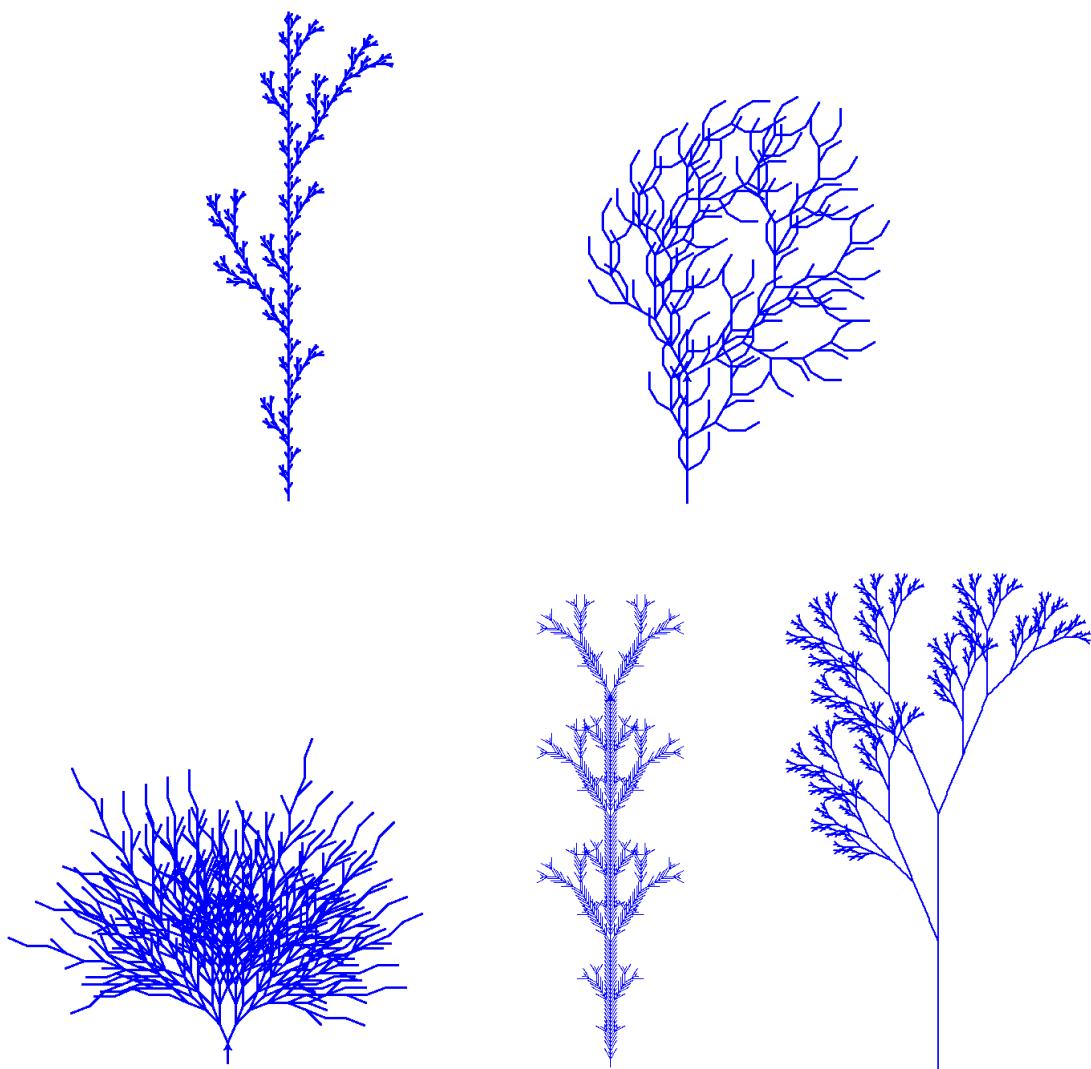


Ci-dessous, à gauche :

initialisation : F règle : $F \rightarrow F[+F]F[-F]F$

Ci-dessous, à droite :

initialisation : F règle : $F \rightarrow FF[-F+F+F]+[+F-F-F]$



De gauche à droite :

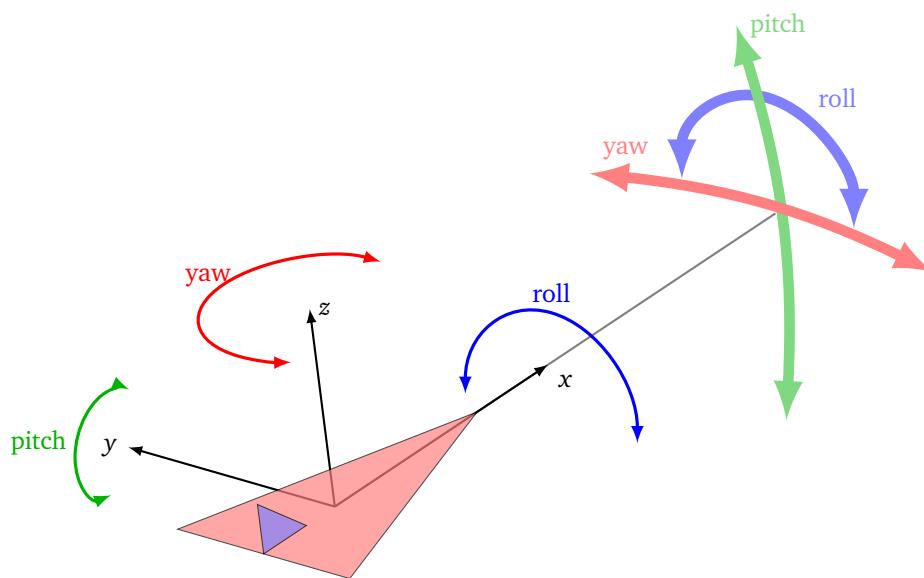
initialisation : F	règle 1 : $F \rightarrow F[-G][+G]$	règle 2 : $G \rightarrow F[-G]F[+F-G]$
initialisation : YYY	règle 1 : $X \rightarrow X[-FFF][+FFF]FX$	règle 2 : $Y \rightarrow YFX[+Y][-Y]$
initialisation : X	règle 1 : $X \rightarrow F[+X]F[-X]+X$	règle 2 : $F \rightarrow FF$

3.4. Plantes en 3D

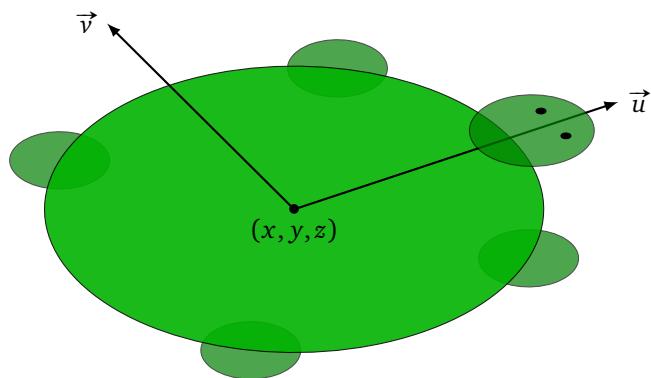
Tortue 3D

Souvenez-vous des trois mouvements de rotation possibles *yaw/pitch/roll* d'un objet de l'espace, par exemple un avion :

- lacet/*yaw* : rotation autour d'un axe haut/bas (un angle positif déplace le nez de l'appareil vers la gauche).
- tangage/*pitch* : rotation autour de l'axe des ailes (un angle positif monte le nez de l'appareil),
- roulis/*roll* : rotation autour de l'axe de la cabine (un angle positif fait monter l'aile gauche et baisser l'aile droite).



Jusqu'ici notre tortue se mouvait dans le plan. Une tortue de l'espace est caractérisée par la position (x, y, z) de son centre, un vecteur \vec{u} pointant vers l'avant et un vecteur \vec{v} indiquant la gauche de la tortue. Le vecteur $\vec{w} = \vec{u} \wedge \vec{v}$, dirigé vers le haut, complète ces vecteurs en un repère orthonormal direct.



On étend le langage de la tortue afin qu'elle puisse se déplacer dans l'espace :

- **F** ou **G** : avance d'une quantité fixée,
- **+** : tourne vers la gauche, sans avancer, d'un angle fixé (lacet à gauche).
- **-** : tourne vers la droite d'un angle fixé (lacet à droite).

- & : baisse le nez d'un angle fixé (tangage vers le bas).
- ^ : monte le nez d'un angle fixé (tangage vers le haut).
- < : effectue un roulis vers la gauche d'un angle fixé.
- > : effectue un roulis vers la droite d'un angle fixé.
- | : fait demi-tour (lacet de 180°).
- [: mémorise la position de la tortue (coordonnées (x, y, z) et vecteurs \vec{u} et \vec{v}).
-] : repart de la position mémorisée par le crochet [correspondant.
- X, Y : ne font rien.

L-système 3D

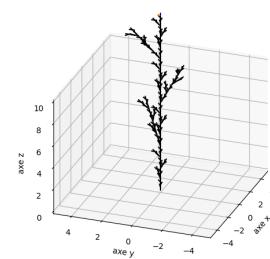
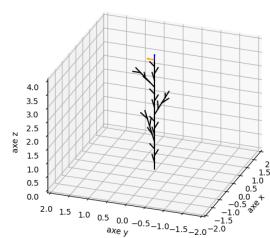
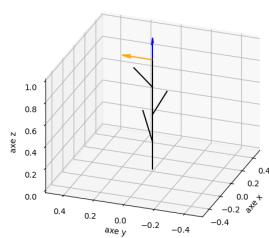


Considérons le L-système :

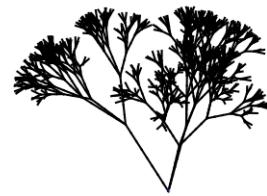
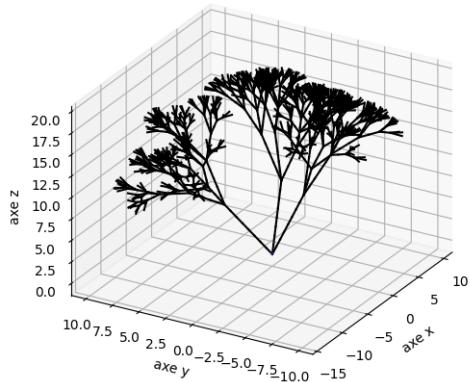
initialisation : F règle : $F \rightarrow F[\&+F]F[->F]F[+^F]F$

L'angle est $\frac{\pi}{8}$.

Voici les tracés des trois premières itérations.



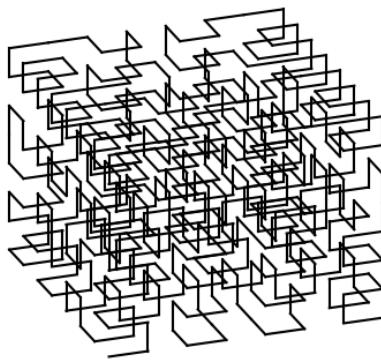
Voici un autre exemple avec deux vues différentes.



Des améliorations sont possibles pour aller plus loin dans le réalisme :

- ajouter une texture et des couleurs,
- diminuer le diamètre et la longueur des branches au fur et à mesure des itérations,
- ajouter des feuilles au bout de chaque branche terminale,
- faire une version stochastique c'est-à-dire décider aléatoirement si une branche doit être dessinée ou pas.

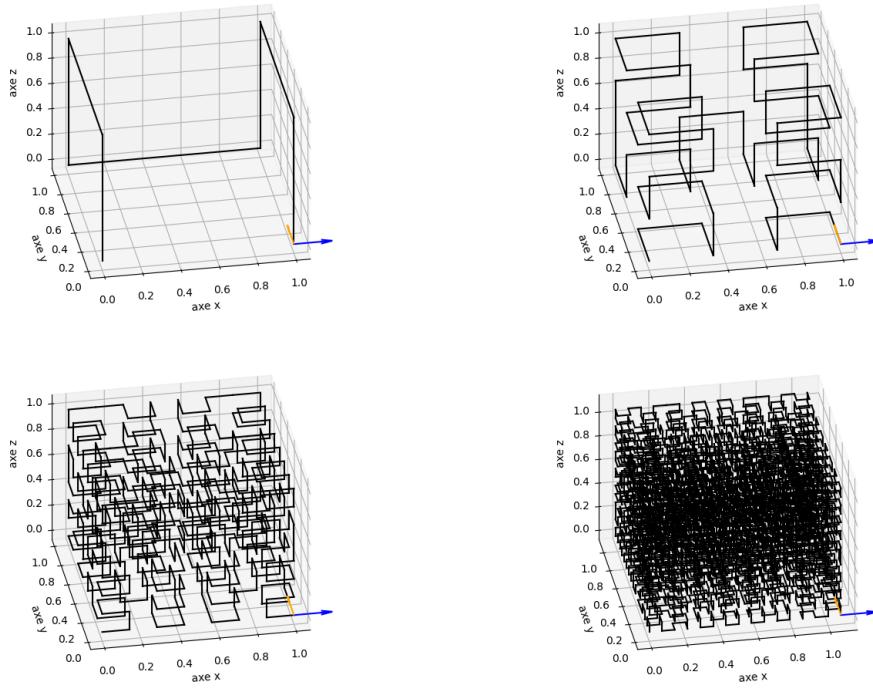
Courbe de Hilbert 3D



Terminons par une courbe continue de l'espace qui remplit complètement le cube unité. Cette courbe peut être approchée par les itérés du L-système :

initialisation : X règle : X → ^ <XF^ <XFX-F^ >>XFX&F+>>XFX-F>X->

L'angle est $\frac{\pi}{2}$.



Cela peut être utile si un personnage ou un robot doit inspecter de près chaque point du volume d'une pièce.

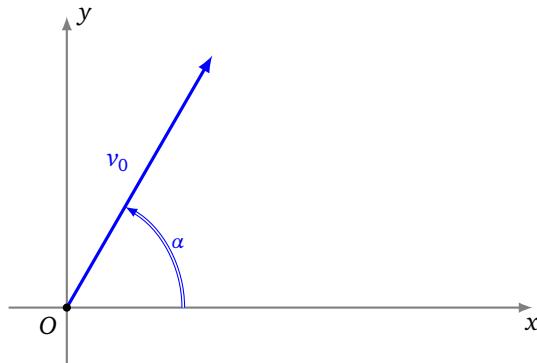
Certains paragraphes sont extraits d'un chapitre « Systèmes itérés de fonctions » d'un cours de géométrie, d'autres du chapitre « L-système » du livre « Python au lycée » dans lesquels vous trouverez plus de détails. Vous trouverez sur notre site une librairie `turtle3d` afin de tracer vos L-systèmes en dimension 3.

Pour rendre des animations réalistes, il faut bien comprendre certains principes issus de la physique. Nous en illustrons quelques-uns.

1. Trajectoire

1.1. Équation

On lance une balle avec une vitesse initiale v_0 et un angle α depuis un point origine O . On note $(x(t), y(t))$ la position de la balle au fil du temps t .



La balle est soumise à son poids \vec{P} et on néglige les frottements. Le principe fondamental de la mécanique nous dit :

$$\sum \vec{F} = m \vec{a}$$

Ici cela devient simplement $m \vec{g} = m \vec{a}$, d'où l'accélération $\vec{a} = \vec{g}$. On obtient les équations différentielles :

$$x''(t) = 0$$

$$y''(t) = -g$$

On impose en plus les conditions initiales :

$$x(0) = 0 \quad y(0) = 0 \quad x'(0) = v_0 \cos \alpha \quad y'(0) = v_0 \sin \alpha$$

Si on intègre une première fois on obtient :

$$x'(t) = v_0 \cos \alpha$$

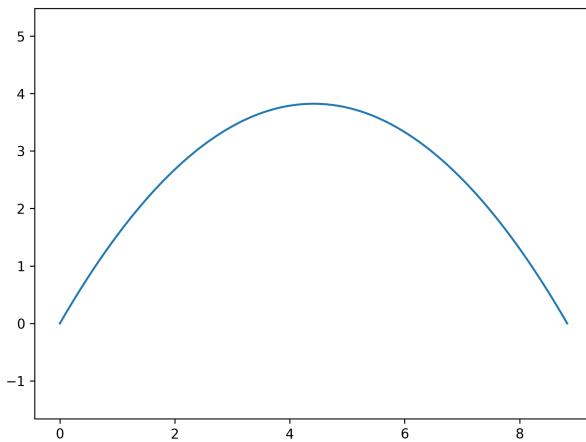
$$y'(t) = v_0 \sin \alpha - gt$$

La trajectoire est alors donnée par :

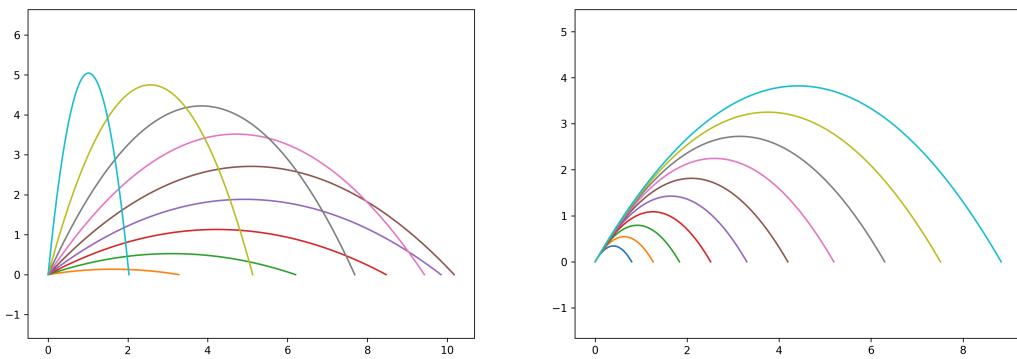
$$x(t) = v_0 \cos(\alpha)t$$

$$y(t) = v_0 \sin(\alpha)t - \frac{1}{2}gt^2$$

Voici la trajectoire pour un lancer avec $\alpha = \frac{\pi}{3}$ et $v_0 = 10$.



Ci-dessous à gauche différents lancers pour différents angles mais une même vitesse initiale ; à droite différentes vitesses initiales pour un même angle.



1.2. Éléments différentiels

On va tracer la trajectoire pas à pas en utilisant les éléments différentiels (comme dans le chapitre « Équations différentielles »). Si $(x(t), y(t))$ est la position de la balle au temps t et $(v_x(t), v_y(t))$ sa vitesse, alors sa position à un temps $t + dt$ est donnée par :

$$\begin{aligned} x(t + dt) &= x(t) + v_x(t)dt \\ y(t + dt) &= y(t) + v_y(t)dt \end{aligned}$$

Mais comment calculer la vitesse ? La vitesse est la dérivée de la position donc $(v_x(t), v_y(t)) = (x''(t), y''(t))$. Comme $x''(t) = 0$ alors $x'(t)$ est constant et par condition initiale $v_x(t) = x'(t) = v_0 \cos \alpha$ est constant. Par contre $y''(t) = -g$, donc $\frac{y'(t+dt)-y'(t)}{dt} = -g$ d'où $y'(t + dt) = y'(t) - gdt$ avec la condition initiale $y'(0) = v_0 \sin \alpha$.

On peut ainsi construire une trajectoire point par point. On pose $(x_0, y_0) = (0, 0)$. On définit un intervalle élémentaire de temps dt (par exemple $dt = 0.01$). La formule de récurrence est alors :

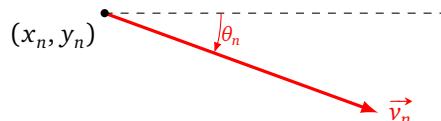
$$\begin{aligned} x_{n+1} &= x_n + v_0 \cos(\alpha)dt \\ y_{n+1} &= y_n + v_0 \sin(\alpha)dt \\ v_{y,n+1} &= v_{y,n} - gdt \end{aligned}$$

Il est rarement facile de pouvoir expliciter les solutions d'une équation différentielle. Par exemple, ajoutons une force de frottement qui s'oppose au mouvement de la balle. Cette force de résistance est donnée en fonction de la vitesse

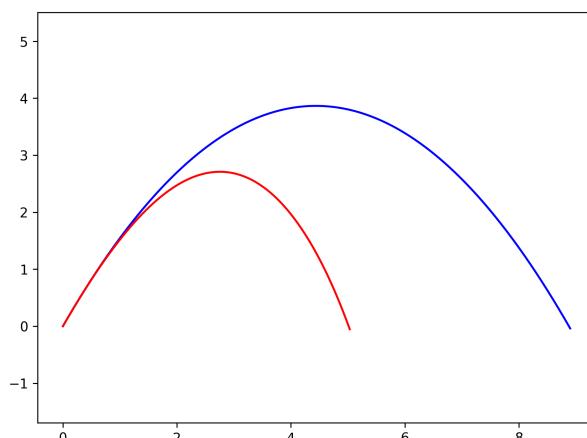
$$\vec{R} = -k \|\vec{v}\|^e \frac{\vec{v}}{\|\vec{v}\|}$$

k est une constante. Des modèles classiques pour l'exposant sont $e = 1$ (la résistance est proportionnelle à la vitesse, valable pour une vitesse faible) ou bien $e = 2$ (la résistance est proportionnelle au carré de la vitesse), des valeurs non entières de e sont aussi intéressantes. Le principe fondamental de la mécanique $\vec{P} + \vec{R} = m\vec{g}$ conduit à des équations différentielles plus compliquées, voire impossibles à résoudre par des formules analytiques. Cependant en utilisant la méthode des éléments différentiels (*aka* la méthode d'Euler) on trace facilement les trajectoires. On pose $(x_0, y_0) = (0, 0)$, $(v_{x,0}, v_{y,0}) = (v_0 \cos \alpha, v_0 \sin \alpha)$. La formule de récurrence est :

$$\begin{aligned}x_{n+1} &= x_n + v_{x,n} dt \\y_{n+1} &= y_n + v_{y,n} dt \\\theta_n &= \arctan2(v_{y,n}, v_{x,n}) \\\|\vec{v}_n\| &= \sqrt{v_{x,n}^2 + v_{y,n}^2} \\v_{x,n+1} &= v_{x,n} - \frac{k}{m} \|\vec{v}_n\|^e \cos \theta_n dt \\v_{y,n+1} &= v_{y,n} - g dt - \frac{k}{m} \|\vec{v}_n\|^e \sin \theta_n dt\end{aligned}$$

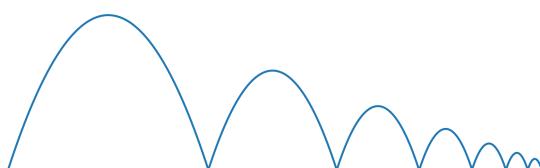


θ_n est l'angle formé par le vecteur vitesse et l'horizontale. Noter que cette fois-ci la masse intervient dans les équations. Ci-dessous une trajectoire avec frottement quadratique ($e = 2$ en rouge) comparée à une trajectoire sans frottement. La courbe avec frottement n'est plus une parabole.



1.3. Rebonds

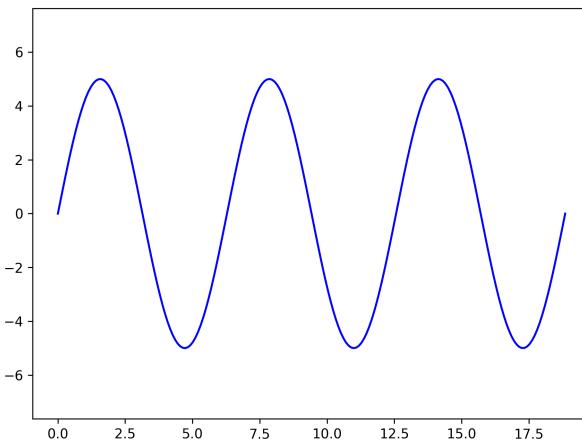
Il est facile de modéliser un rebond. Par exemple dès que l'ordonnée y de la balle devient négative, on simule un nouveau lancer de balle. Afin de modéliser la perte d'énergie au fur et à mesure des rebonds, la vitesse du lancer au rebond numéro k diminue à chaque rebond selon une formule $v_{k+1} = cv_k$, avec une vitesse initiale v_0 et un coefficient $0 \leq c \leq 1$ (par exemple $c = 0.8$).



2. Ondes

2.1. Une dimension

Une onde est le déplacement d'une perturbation d'une propriété physique par rapport à son état d'équilibre. Par exemple une onde sonore est une modification de la pression de l'air : des particules d'air sont comprimées par la source sonore, elles vont perturber les particules voisines qui à leur tour perturbent d'autres particules.



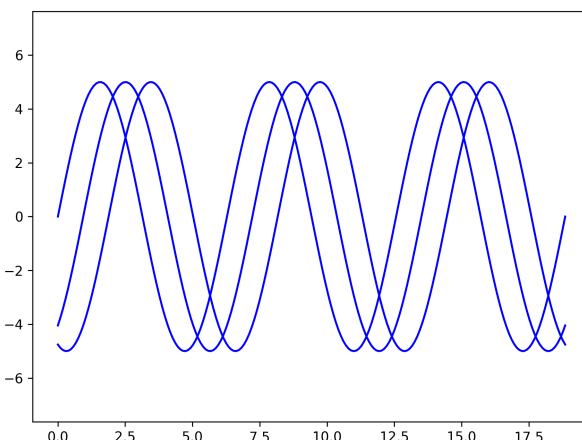
Beaucoup de phénomènes, comme des vagues par exemple, sont modélisés par des ondes sinusoïdales. L'équation est

$$F(x, t) = A \sin\left(\frac{2\pi}{\lambda}x - \frac{2\pi}{T}t\right)$$

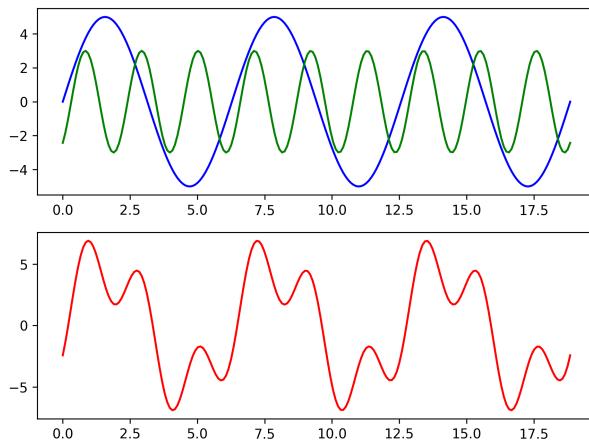
où

- $F(x, t)$ représente une propriété physique,
- en fonction d'une position $x \in \mathbb{R}$,
- et d'un instant t ,
- A est l'amplitude,
- λ est la longueur d'onde,
- T est la période.

La longueur d'onde λ mesure la distance (en mètre) entre deux crêtes d'une même sinusoïde. L'onde se décale vers la droite au fil du temps, son graphe se superpose à lui-même au bout de T secondes. La vitesse de l'onde se calcule par $v = \frac{\lambda}{T}$.

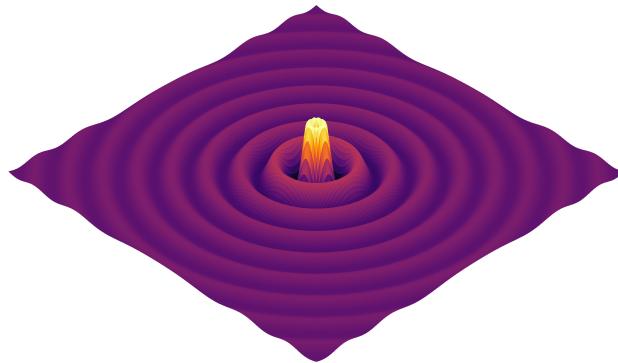


Lorsque l'on superpose deux ondes sinusoïdales $F_1(x, t) + F_2(x, t)$ ayant des caractéristiques différentes, on obtient des fonctions plus compliquées. Ci-dessous les graphes de deux ondes F_1 et F_2 (en haut) ainsi que leur superposition $F_1 + F_2$ (en bas).



2.2. Deux dimensions

Lorsque qu'une pierre tombe dans l'eau, cela entraîne une onde à la surface sous la forme de cercles qui s'agrandissent. Par contre, la hauteur des vagues diminue lorsque l'on s'éloigne de la source car une même énergie se répartit dans toutes les directions.

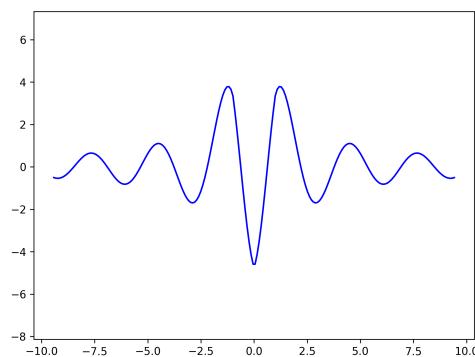


La formule correspondante est :

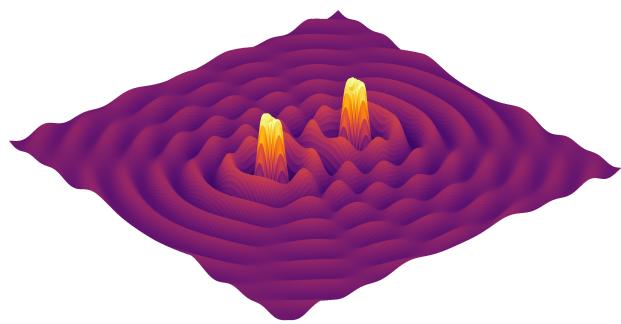
$$F(r, t) = \frac{A}{r} \sin\left(\frac{2\pi}{\lambda} r - \frac{2\pi}{T} t\right)$$

où $r = \sqrt{x^2 + y^2}$ mesure la distance à l'origine.

Voici l'allure d'une tranche selon le plan ($y = 0$).

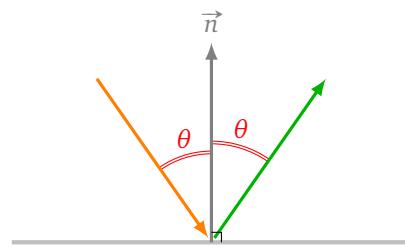


Il est intéressant de lancer deux pierres en même temps vers des points différents. Les ondes se superposent et on voit apparaître des interférences, par exemple des zones où la hauteur des vagues est nulle (la crête issue de la première pierre est compensée par le creux issu de la seconde pierre et réciproquement).

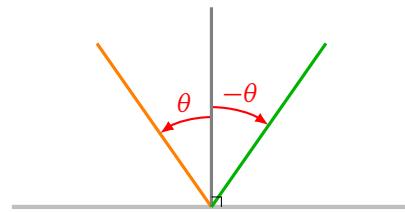


2.3. Réflexion

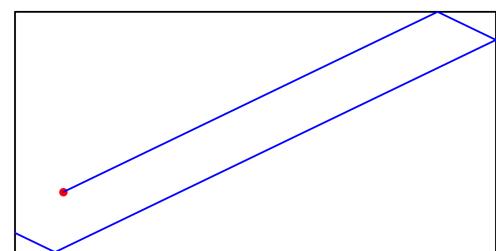
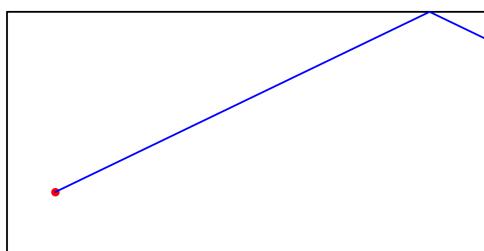
Un rayon lumineux se réfléchit sur un miroir selon la loi de réflexion : l'angle entre la normale et le rayon incident est égal à l'angle entre la normale et le rayon réfléchi. On renvoie aux chapitres « Lancer de rayons » et « Lumière » pour plus de détails.

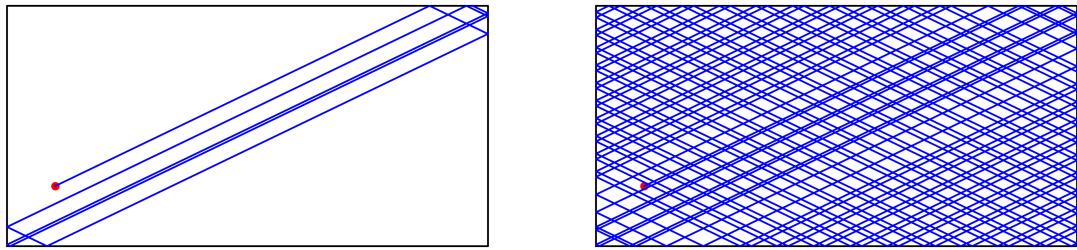


En fait, pour être plus précis, on devrait parler d'angles orientés et alors les angles incidents et réfléchis sont opposés.



Considérons l'exemple suivant : on fixe un point initial et une direction à partir desquels on lance un rayon qui se réfléchit sur les quatre bords d'un rectangle. On trace alors sa trajectoire. Ci-dessous : 1, 2, 10, 100 réflexions.





Pour les calculs, le plus simple est de considérer le rayon paramétré par :

$$\begin{cases} x = x_0 + t \cos \theta \\ y = y_0 + t \sin \theta \end{cases}$$

On calcule facilement chaque paramètre t_h , t_b , t_d , t_g qui correspond à l'intersection de chacune des faces haut/bas/droite/gauche. Par exemple, on obtient t_h en résolvant l'équation $y_{\text{haut}} = y_0 + t \sin(\theta)$ (où y_{haut} correspond à l'ordonnée de la face en haut). La face sur laquelle le rayon se réfléchit correspond au paramètre $t > 0$ le plus petit parmi t_h , t_b , t_d , t_g . Si par exemple le rayon se réfléchit sur la face haute alors on repart du point de cette face mais cette fois avec un angle $-\theta$.

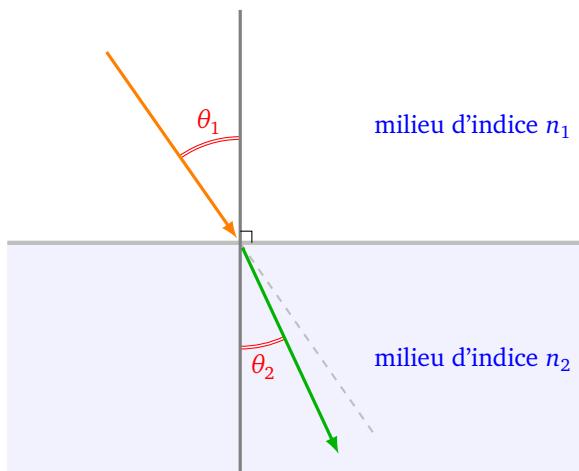
2.4. Réfraction

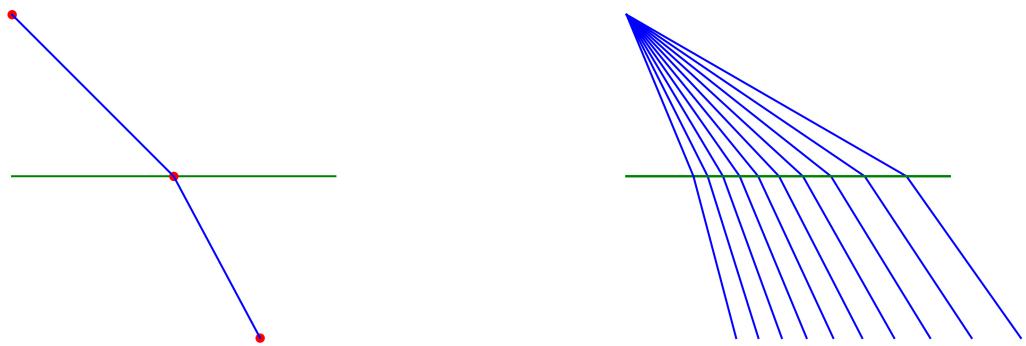
Lorsque un rayon lumineux change de milieu, par exemple passe de l'air à l'eau, la trajectoire du rayon est modifiée : c'est un phénomène de réfraction.

La **loi de Snell-Descartes** relie l'angle entre le rayon incident avec la normale et le rayon réfracté et la normale :

$$n_1 \sin \theta_1 = n_2 \sin \theta_2$$

n_1 et n_2 sont les indices des milieux correspondants, par exemple l'indice de l'air est environ $n_{\text{air}} \simeq 1$ et celui de l'eau est $n_{\text{eau}} \simeq 1.33$.



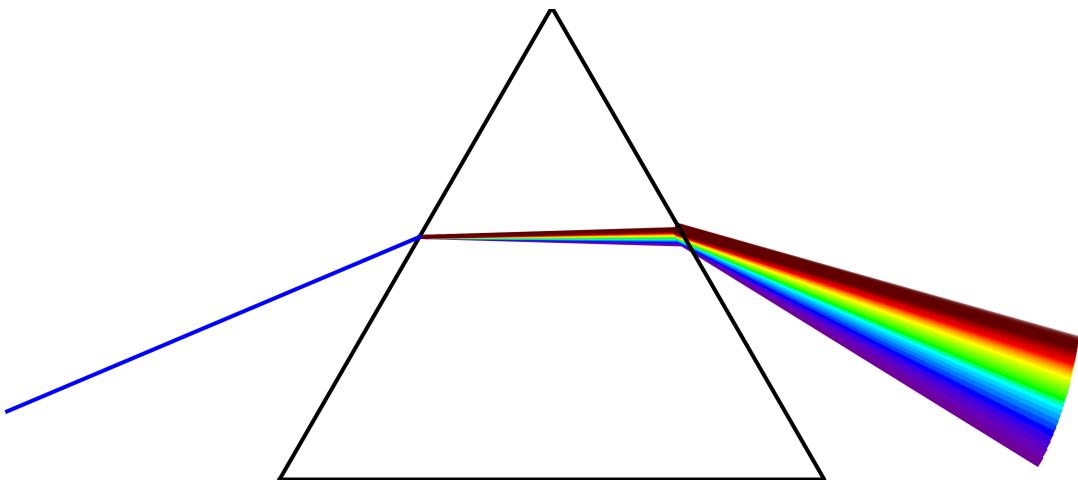


L'indice du verre est environ $n_{\text{verre}} \simeq 1.5$. Mais en fait l'indice précis dépend de la longueur d'onde du rayon lumineux. Un rayon de lumière blanche est la superposition de rayons lumineux de différentes longueurs d'onde (du violet de longueur d'onde démarrant à 380 nanomètres au rouge terminant vers 780 nm). Comme l'indice du verre dépend de la longueur d'onde, le rayon de lumière blanche va se séparer en différents rayons colorés en traversant la surface du verre. L'expérience de Newton avec un prisme de verre met ainsi en évidence le spectre de la lumière blanche.

Une formule approchée, dite de Cauchy, pour l'indice du verre est

$$n_{\text{verre}}(\lambda) = 1.4580 + \frac{0.0354}{\lambda^2}$$

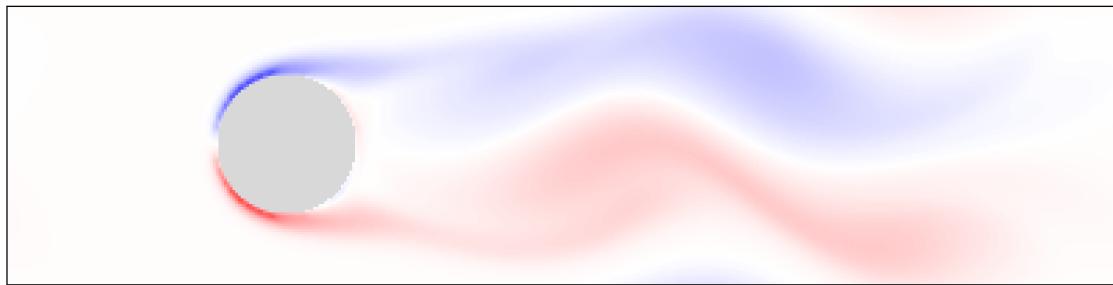
où λ est la longueur l'onde du rayon incident exprimée en micromètre (donc variant de 0.380 à 0.780). C'est un bon exercice de programmation de simuler cette expérience fondamentale.



3. Liquide et gaz

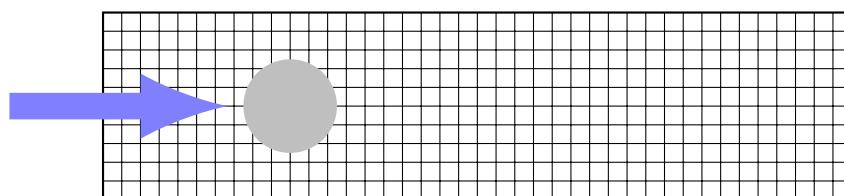
La modélisation des fluides est extrêmement complexe. La mécanique des liquides et des gaz est régie par des lois approximant le comportement fluide en supposant des propriétés simplificatrices (par exemple fluide sans viscosité ou bien incompressible...). Même avec ces hypothèses, les équations qui régissent le comportement sont des équations différentielles (plus précisément des équations aux dérivées partielles) dont on ne sait en général pas calculer des solutions explicites. Les équations simples (fluide sans viscosité) sont les équations d'Euler, le cas général est régi par les équations de Navier-Stokes (dont personne ne connaît de solution générale et qui font l'objet de recherches actives).

Nous allons modéliser le déplacement d'un fluide incompressible dans un tube lorsqu'il rencontre un obstacle (ici un disque). Cette section est basée sur l'article *Create your own lattice Boltzmann simulation* de Philip Mocz.

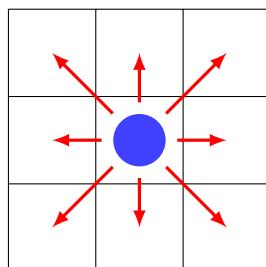


3.1. Réseau de Boltzmann

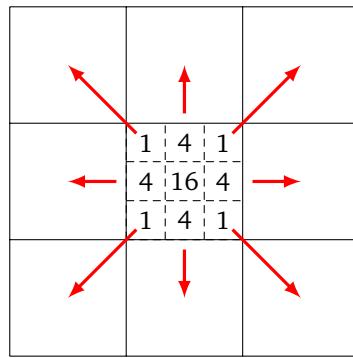
Commençons par décrire le comportement à l'équilibre d'un liquide ou d'un gaz par une modélisation microscopique d'une seule particule. Même si le fluide est globalement à l'équilibre chaque particule peut se déplacer de façon aléatoire. Nous modélisons le lieu d'évolution par une grille rectangulaire de taille $N_x \times N_y$ dans laquelle on a placé un obstacle (ici un disque).



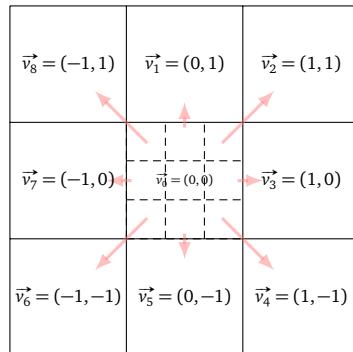
Dans un premier temps on considère que dans chaque case (x, y) on place une particule ; à l'instant $t + dt$ suivant cette particule peut se déplacer sur l'une des 8 cases voisines ou bien rester à sa place.



Comme on ne sait pas quel va être le comportement de cette particule et qu'on veut considérer toutes les possibilités on remplace cette grosse particule dans la case (x, y) par 36 mini-particules. Ces 36 mini-particules représentent toutes les possibilités de déplacement en tenant compte des probabilités : 16 mini-particules resteront immobiles, 4 mini-particules iront vers la case de droite, 4 vers la case de gauche, ..., 1 seule mini-particule sur les 36 ira sur la case en haut à droite, ...



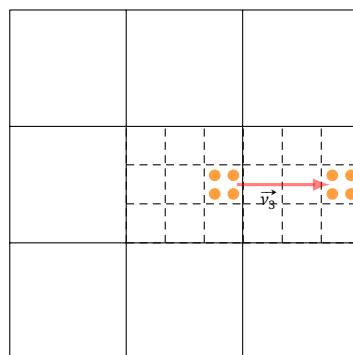
On associe donc à chaque mini-particule en (x, y) un vecteur vitesse \vec{v}_i parmi $\vec{v}_0 = (0, 0)$ (la mini-particule reste immobile), $\vec{v}_1 = (0, 1)$ (la mini-particule va monter d'une case), ..., $\vec{v}_8 = (-1, 1)$.



Ainsi l'ensemble des mini-particules est modélisé par une fonction $F(x, y, i)$ qui compte le nombre de particules dans la case (x, y) ayant pour vecteur vitesse \vec{v}_i . Par exemple $F(x, y, 1)$ compte le nombre de mini-particules sur la case (x, y) qui à l'instant suivant se déplaceront vers la case $(x, y + 1)$.

3.2. Mouvement

Comment modéliser le déplacement ? C'est tout simple : chaque mini-particule se déplace en suivant son vecteur vitesse. Ainsi les mini-particules sur la case (x, y) avec le vecteur \vec{v}_i se déplacent à l'instant suivant vers la case $(x, y) + \vec{v}_i$. En plus, par inertie, elles conservent leur vecteur vitesse \vec{v}_i .



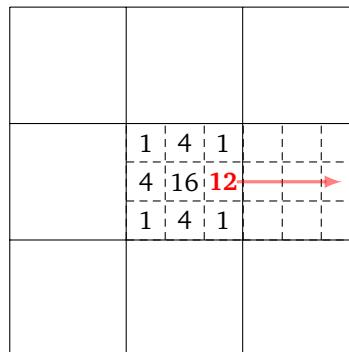
Sur une case (x, y) des mini-particules peuvent arriver de plusieurs directions, mais si on impose en plus le vecteur vitesse alors les mini-particules ne proviennent que d'une seule case. D'où la formule :

$$F(x + v_{x,i}, y + v_{y,i}, i) = F(x, y, i)$$

où $\vec{v}_i = (v_{x,i}, v_{y,i})$.

3.3. Flux vers la droite

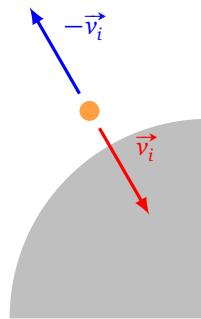
Pour l'instant depuis une case les mini-particules se déplacent vers la droite ou vers la gauche de manière équiprobable. Si on veut modéliser un flux vers la droite (par exemple l'eau d'une rivière qui s'écoule de la gauche vers la droite, ou bien l'action d'un ventilateur ou d'une turbine) il suffit d'ajouter des mini-particules ayant le vecteur vitesse $\vec{v}_3 = (1, 0)$. Par exemple on y place 12 mini-particules au lieu des 4 de l'état d'équilibre.



Pour éviter un modèle trop « lisse » on perturbe aléatoirement l'initialisation : on ajoute au hasard, ici où là, une mini-particule à l'état (x, y, i) .

3.4. Obstacle

Que fait la particule lorsqu'elle rencontre un obstacle ? Elle repart simplement en sens inverse. C'est une approximation grossière dans laquelle on considère que nos mini-particules arrivent perpendiculairement au bord. Plus en détails : si une particule (x, y) de vecteur \vec{v}_i devait se retrouver à la prochaine itération dans l'obstacle, au lieu de se retrouver en $(x, y) + \vec{v}_i$, on la conserve en (x, y) mais avec le vecteur $-\vec{v}_i$.

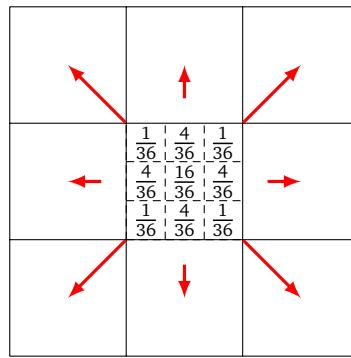


3.5. Densité et vitesse

La *densité* ρ en (x, y) est le nombre de mini-particules dans la case (x, y) , elle s'obtient en comptant les mini-particules en (x, y) quel que soit leur vecteur vitesse :

$$\rho(x, y) = F(x, y, 0) + F(x, y, 1) + \cdots + F(x, y, 8).$$

Lors de l'initialisation il est préférable de supposer que la densité vaut 1 en tout (x, y) quitte à faire une étape de renormalisation. Par exemple, voici une case (x, y) de densité 1 qui correspond à un état d'équilibre (sans flux) : une mini-particule correspond ici à 1/36 d'une particule.



Par contre au fil du temps la densité va évoluer et ne vaut plus 1 partout.

Une autre valeur physique importante est le *vecteur vitesse* $\vec{u}(x, y)$ en (x, y) . On calcule ce vecteur vitesse en le décomposant sous la forme (vitesse horizontale, vitesse verticale) : $\vec{u} = (u_x, u_y)$. La *vitesse horizontale* est le nombre de mini-particules allant vers la droite moins le nombre de mini-particules allant vers la gauche divisé par la densité :

$$u_x(x, y) = \frac{1}{\rho(x, y)}(F(x, y, 2) + F(x, y, 3) + F(x, y, 4) - F(x, y, 6) - F(x, y, 7) - F(x, y, 8)).$$

La *vitesse verticale* est le nombre de mini-particules montantes moins le nombre de mini-particules descendantes (divisé par le nombre total de mini-particules) :

$$u_y(x, y) = \frac{1}{\rho(x, y)}(F(x, y, 1) + F(x, y, 2) - F(x, y, 4) - F(x, y, 5) - F(x, y, 6) + F(x, y, 8)).$$

3.6. Collision et fonction d'équilibre

Voici les trois étapes lors de chaque itération correspondant à un intervalle de temps élémentaire dt :

1. *Mouvement* : chaque mini-particule se déplace suivant son vecteur vitesse \vec{v}_i : $F(x + v_{x,i}, y + v_{y,i}, i) = F(x, y, i)$.
2. *Obstacle* : les mini-particules qui rencontrent l'obstacle font demi-tour, pour elles \vec{v}_i devient $-\vec{v}_i$.
3. *Collision* : on modifie la densité F afin de tenir compte des collisions entre particules, cela se fait à l'aide d'une fonction d'équilibre F_{eq} et on remplace F par $F - \frac{1}{\tau}(F - F_{eq})$.

Détaillons cette dernière étape. Tout d'abord la fonction d'équilibre est issue de la physique et des équations de Navier-Stokes :

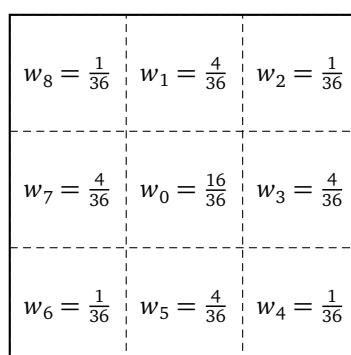
$$F_{eq}(x, y, i) = w_i \rho(x, y) \left(1 + 3\vec{u} \cdot \vec{v}_i + \frac{9}{2}(\vec{u} \cdot \vec{v}_i)^2 - \frac{3}{2}\|\vec{u}\|^2 \right).$$

C'est-à-dire :

$$F_{eq}(x, y, i) = w_i \rho(x, y) \left(1 + 3(u_x v_{x,i} + u_y v_{y,i}) + \frac{9}{2}(u_x v_{x,i} + u_y v_{y,i})^2 - \frac{3}{2}(u_x^2 + u_y^2) \right).$$

On a abrégé $u_x(x, y)$, $v_{x,i}(x, y)$... en u_x , $v_{x,i}$...

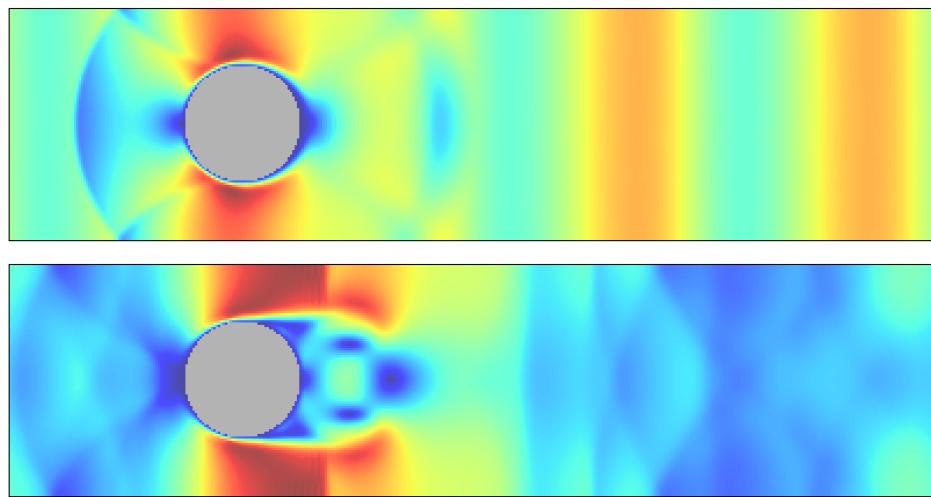
Les w_i ($i = 0, \dots, 8$) sont les poids associés à la position d'équilibre statique $w_0 = \frac{16}{36}$, $w_1 = \frac{4}{36}$, ...



Pour tenir compte des collisions entre particules on mesure la différence entre l'état actuel et l'état à l'équilibre, que l'on pondère par un coefficient $\frac{1}{\tau}$ (où τ est le temps de collision, par exemple $\tau = 0.6$). On remplace $F(x, y, i)$ par $F(x, y, i) - \frac{1}{\tau}(F(x, y, i) - F_{\text{eq}}(x, y, i))$, autrement dit on définirait par récurrence $F_{n+1}(x, y, i) = F_n(x, y, i) - \frac{1}{\tau}(F_n(x, y, i) - F_{\text{eq}}(x, y, i))$.

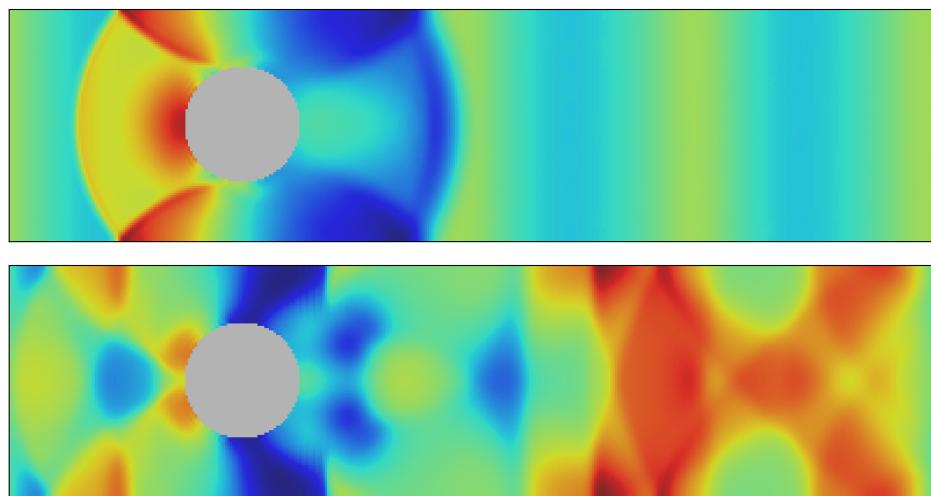
3.7. Résultats

Voici quelques images illustrant les résultats. Ci-dessous la mesure de la vitesse $\|\vec{u}(x, y)\|$ des particules après 100 itérations (en haut) et 500 (en bas). On remarque que les particules sont plus rapides en-dessous et au-dessus de l'obstacle (en rouge) et plus lentes avant et après l'obstacle (en bleu). Cela reflète bien que les particules sont bloquées par l'obstacle et aussi que le flux doit accélérer au niveau de l'obstacle car le passage y est moins large.



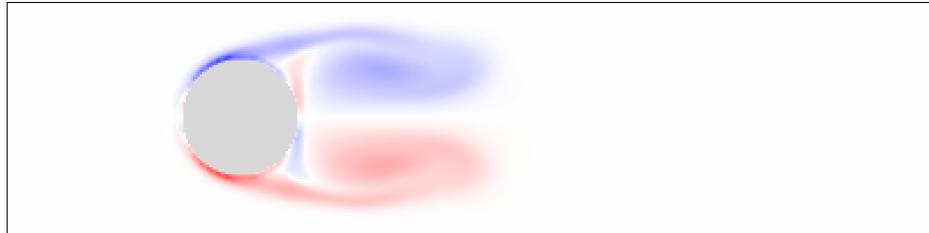
Il faut noter que dans toutes ces simulations, on choisit de réinjecter par la gauche, le flux sortant du rectangle vers la droite. Ainsi la perturbation générée par l'obstacle finit par se faire ressentir devant l'obstacle après un certain nombre d'itérations.

Ci-dessous la mesure de la densité $\rho(x, y)$ des particules après 100 itérations (en haut) et 500 (en bas). On note une densité plus forte devant l'obstacle qui s'explique par le rebond.



3.8. Vortex

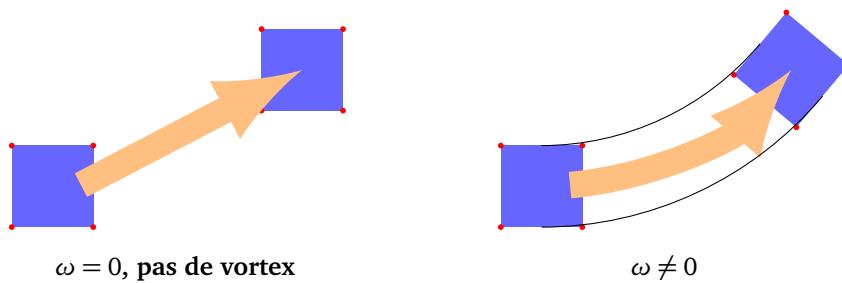
Les plus belles images sont obtenues en calculant l'effet vortex (*vorticity*). Ci-dessous le vortex pour 1000 itérations. Les particules qui tournent vers la droite en se déplaçant sont en bleu. Les particules qui tournent vers la gauche en se déplaçant sont en rouge.



Le **vortex** se calcule selon la formule :

$$\omega(x, y) = \frac{\partial u_y}{\partial x} - \frac{\partial u_x}{\partial y}.$$

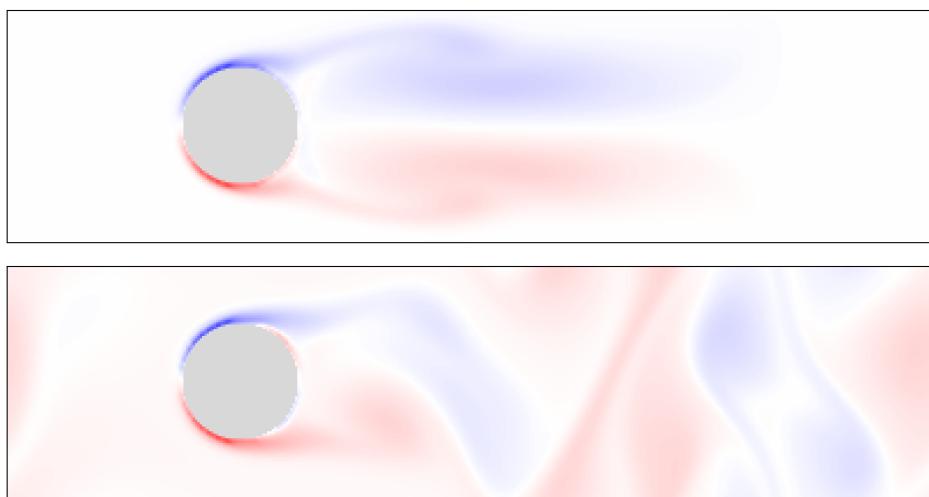
Le vortex mesure comment un petit ensemble de particules tourne par rapport à son déplacement. Ci-dessous un petit carré qui contient plusieurs particules. Sur la figure de gauche il n'y a pas de vortex ($\omega = 0$), à droite le carré tourne en se déplaçant, le vortex y est non-nul.



Pour calculer une version discrète du vortex on effectue :

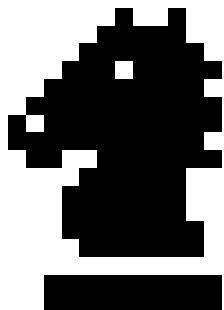
$$\omega = (u_y(x+1, y) - u_y(x-1, y)) - (u_x(x, y+1) - u_x(x, y-1)).$$

Ci-dessous la situation au bout de 2000 et 4000 itérations.



Référence : [Create your own lattice Boltzmann simulation](#) et [page github](#) de Philip Mocz. La fonction d'équilibre qui y est présentée doit être corrigée en $F_{eq}(x, y, i) = w_i \rho (1 + 3 \vec{u} \cdot \vec{v}_i + \frac{9}{2} (\vec{u} \cdot \vec{v}_i)^2 - \frac{3}{2} \|\vec{u}\|^2)$ comme énoncée ci-dessus.

QUATRIÈME PARTIE



THÉORIE DES JEUX

Théorie des jeux

Nous survolons les différents types de jeux, leurs caractéristiques, les différentes stratégies possibles et les équilibres possibles entre adversaires.

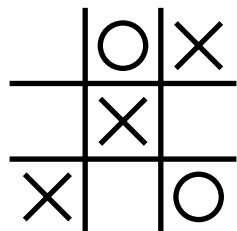
1. Quelques jeux

1.1. Motivation

Nous allons étudier surtout des jeux avec deux joueurs (comme les échecs). Les jeux avec plus de joueurs sont plus délicats car les alliances possibles compliquent les stratégies, pensez au jeu « poules/renards/vipères » où les renards doivent attraper les poules qui, elles, doivent attraper les vipères qui, elles, doivent attraper les renards !

La théorie des jeux a de nombreuses applications. Tout d'abord à l'économie, par exemple les entreprises concurrentes A et B qui fabriquent le même produit sont comme les deux adversaires d'un jeu. Chaque entreprise a à sa disposition plusieurs stratégies possibles (type de produits, qualité, prix, innovation...). Le gain du jeu étant ici des revenus ou des parts de marché. De même l'acheteur et le vendeur ont des intérêts opposés lorsqu'il s'agit de fixer le prix. De nombreux mathématiciens ont obtenu la récompense dite « prix Nobel d'économie » pour leurs travaux en théorie des jeux, le plus célèbre étant John Nash (1928–2015). Une autre application concerne les relations internationales : comment éviter ou résoudre un conflit entre deux pays dont les intérêts sont opposés car ils convoitent les mêmes ressources ou le même territoire ?

Voici quelques jeux classiques que vous connaissez sûrement : échecs, dames, go, reversi/othello, puissance 4..., mais aussi des jeux avec du hasard comme le backgammon, le monopoly, ou les jeux de cartes... Un autre grand classique est le jeu du morpion (*tic-tac-toe*) dans lequel il s'agit d'aligner trois croix ou trois ronds sur une grille 3×3 . C'est un jeu vite lassant mais intéressant pour apprendre à programmer et tester des algorithmes.



1.2. Matrice de gains

Nous allons découvrir un nouveau type de jeu qui permet de modéliser de nombreuses situations. La version la plus connue de ce jeu est le « dilemme du prisonnier » : un juge interroge, chacun leur tour, deux hommes A et B soupçonnés d'un cambriolage pour lequel il n'y a aucune preuve, ni témoin. Il propose à A soit d'être solidaire de son complice en se taisant, soit de le trahir en l'accusant. Le juge propose ensuite séparément à B les mêmes choix. A et B ne peuvent pas se concerter. Le juge les condamne alors selon leurs réponses à tous les deux :

- si chacun est solidaire de son complice et tous les deux nient leur participation, le juge les condamnera à 1 an de prison chacun,
- si les deux trahissent et s'accusent mutuellement alors le juge les condamnera à 3 ans de prison chacun,
- si A est solidaire mais que B le trahit alors le juge condamne A à 5 ans de prison et B est libre,
- si B est solidaire mais que A le trahit alors le juge condamne B à 5 ans de prison et A est libre.

On résume cela sous la forme d'un tableau :

		Joueur B	
		S	T
Joueur A	S	1 1	0 5
	T	0 5	3 3

Ce jeu est intéressant car si on se place d'un point de vue extérieur il est clair que les prisonniers ont intérêt à être solidaires car dans ce cas ils sont condamnés à un total de 2 ans de prison (1 an chacun). Par contre si on prend le point de vue du prisonnier A par exemple, alors il a intérêt à trahir car il sera alors condamné à 3 ans ou bien libre, alors que s'il est solidaire il risque d'être condamné à 5 ans.

Note : ce tableau s'appelle la **matrice des gains** et le but de chaque joueur est d'obtenir la valeur la plus élevée, comme ici nos prisonniers veulent la peine la plus courte possible il faut prendre les valeurs opposées :

		Joueur B	
		S	T
Joueur A	S	-1 -1	0 -5
	T	-5 0	-3 -3

En changeant les valeurs de la matrice des gains on obtient plusieurs variantes.

Voici le cas général.

		Joueur B	
		S1	S2
Joueur A	S1	a' a	b' b
	S2	c' c	d' d

Tir au but. Un tireur et un gardien de but s'affrontent. Le tireur tire d'un côté (gauche ou droite) et le gardien part d'un côté (gauche ou droite, du point de vue du tireur). Si le gardien part du côté opposé, le tireur marque le but, si le gardien part du bon côté il empêche le but. Voici la matrice des gains.

	Gardien	
	G	D
Tireur	G	1 1
	D	0 1

Pierre/feuille/ciseau. Il y a trois stratégies possibles pour chaque joueur : P/F/C.

	Joueur B			
	P	F	C	
Joueur A	P	0 0	1 1	0
	F	0 1	0 0	1
	C	1 0	0 1	0

Jeux du cerf. Deux chasseurs choisissent sans se concerter si chacun part chasser un lièvre, ou s'il préfère chasser un cerf. Un cerf apporte un plus gros gain, mais ne peut se chasser qu'à deux.

	Joueur B		
	Cerf	Lièvre	
Joueur A	Cerf	2 0	1
	Lièvre	0 1	1

Ce dernier jeu s'appelle aussi jeu d'assurance, par exemple un pays n'a pas d'intérêt à lutter seul contre le changement climatique : cela lui coûte cher et ne produira que peu d'effets si les autres pays refusent d'agir.

1.3. Répétition

Certains jeux ont davantage d'intérêt lorsqu'ils sont répétés. Par exemple pour le jeu du tir au but, « pierre/feuille/ciseaux » ou pour le poker, on peut adapter sa stratégie en fonction des parties précédentes. Le gardien de but peut décider de partir du côté du tir précédent, ou alors il peut anticiper que le tireur va changer de côté et donc le gardien plonge de l'autre côté, mais le tireur peut faire le raisonnement inverse... Une autre stratégie possible est de prendre chaque décision au hasard (droite/gauche comme pile ou face).

Voyons un jeu « *tit-for-tat* » (qu'on pourrait traduire par « donnant-donnant »). La situation est la suivante : vous avez un voisin bruyant et certains soirs il met la musique à fond ; en rétorsion vous pouvez aussi mettre la musique à fond ou bien vous pouvez ne rien faire. La matrice des gains est la suivante :

	Joueur B		
	Silence	Bruit	
Joueur A	Silence	2 0	3
	Bruit	0 1	1

- si vous mettez de la musique tous les deux, vous êtes tous les deux dans une situation de match nul (1 point chacun),
- si aucun de vous ne met de la musique, la situation est gagnant-gagnant (2 points chacun),
- si l'un met la musique et l'autre pas alors la situation est gagnant-perdant (3 points pour celui qui met la musique, 0 pour celui qui ne la met pas).

C'est par exemple une situation que l'on retrouve dans les relations internationales avec les droits de douane : si un pays A impose des droits de douanes sur des produits en provenance du pays B alors généralement ce dernier fait de même sur des produits venant de A.

Le jeu se jouant chaque jour, quelle est la meilleure stratégie à adopter en fonction des jours précédents ? Voici plusieurs stratégies possibles :

- bisounours : ne jamais faire de bruit quelle que soit l'attitude du voisin,
- méchant : toujours faire du bruit quelle que soit l'attitude du voisin,
- vengeance : ne pas faire de bruit au début, mais dès que le voisin fait du bruit, faire du bruit toutes les nuits suivantes,
- cyclothymique : faire du bruit ou pas aléatoirement,

Il peut être amusant de programmer un tel tournoi sur ordinateur, où chaque joueur propose une stratégie qui est ensuite confrontée à toutes les autres stratégies. Les expériences montrent que la meilleure stratégie est « coopération-réciprocité-pardon » qui comporte trois phases :

- *coopération* : au début je ne fais pas de bruit,
- *réciprocité* : ensuite si mon voisin a fait du bruit la nuit précédente, je fais du bruit la nuit suivante ; s'il n'a pas fait de bruit, je n'en fais pas non plus la nuit suivante,
- *pardon* : même si mon voisin a fait du bruit plusieurs nuits de suite, je fais quelques nuits sans bruit afin de revenir à une phase de coopération.

2. Types de jeux

Nous décrivons différentes caractéristiques des jeux.

2.1. Coopératif/non coopératif

Un jeu est **coopératif** si les joueurs décident ensemble de ce qu'il faut faire pour gagner. Généralement ils luttent contre un adversaire commun (la nature, le chronomètre, le réchauffement climatique...). Les joueurs n'ont pas forcément tous les mêmes caractéristiques ni les mêmes stratégies à leur disposition (par exemple deux pays qui luttent contre une menace commune).

Dans un jeu **non coopératif** chaque joueur décide selon son propre intérêt. Dans la plupart des jeux, son intérêt est opposé à celui de ses adversaires (échec, football, poker...), mais pas toujours (par exemple avec plusieurs joueurs il peut être intéressant de créer des alliances). Le dilemme du prisonnier est un jeu non coopératif dans lequel les intérêts des deux joueurs ne sont pas complètement opposés. Noter que si les deux prisonniers pouvaient se mettre d'accord avant de voir le juge, alors le jeu deviendrait coopératif et la meilleure stratégie commune serait que les deux soient solidaires.

2.2. Coups alternés/simultanés

Un jeu peut se dérouler en **coups alternés** (un joueur joue, puis le suivant..., comme les échecs, la belote, le tennis) ou bien en **simultané** (le dilemme du prisonnier, le tireur et le gardien de but qui prennent leur décision en même temps, les jeux vidéos en temps réel). Pour les jeux en simultané, les joueurs basent généralement leur stratégie sur l'historique des coups précédents. Par exemple l'entreprise A ne connaît pas les caractéristiques du prochain téléphone fabriqué par son concurrent B, par contre au vu des évolutions

précédentes elle se doute que le processeur de B sera encore plus rapide et l'appareil photo encore plus précis, elle adapte sa stratégie en conséquence.

2.3. Aléa

Un jeu qui contient une part de hasard est un jeu **aléatoire**. Des exemples de jeux complètement aléatoires sont la roulette, le loto, le jeu pile ou face. Certains jeux ont une part d'aléatoire comme le backgammon, le monopoly et plus généralement les jeux avec des dés ou des cartes à distribuer. Certains jeux, comme les échecs, n'ont aucune part de hasard.

Dans certains cas c'est la stratégie du joueur elle-même qui peut être aléatoire, par exemple le tireur au but peut décider aléatoirement de tirer vers la gauche ou la droite, cela empêche le gardien de faire une prédiction en fonction des tirs précédents. Il faut noter que les humains sont assez mauvais à générer des séries aléatoires : si vous demandez à quelqu'un d'inventer une série de pile/face au hasard, ce sera une série aux caractéristiques différentes d'un vrai tirage aléatoire, par exemple un humain ne dira jamais « pile-pile-pile » car après 2 « piles » on pense inconsciemment que « face » a plus de chances d'arriver ! Les ordinateurs sont bien meilleurs pour produire des séries qui simulent très bien le hasard, on parle de générateur « pseudo-aléatoire ».

2.4. Jeux à somme nulle

Un **jeu à somme nulle** est un jeu où lorsqu'un joueur gagne un point, son adversaire perd un point. C'est par exemple le cas des échecs, si je prends un pion à l'adversaire, il perd un pion ! C'est aussi le cas des paris : si je gagne 1 euro en misant sur le bon cheval alors cet euro vient d'une personne ayant misé sur un mauvais cheval. Un jeu à somme nulle est caractéristique d'un jeu où les ressources sont finies : si je veux quelque chose alors je dois le prendre à quelqu'un.

Voici des exemples de jeu à somme non nulle : les jeux coopératifs (tout le monde gagne ou bien tout le monde perd) ou le dilemme du prisonnier. Cela peut aussi être appliqué en économie : je t'achète le produit A que je ne sais pas fabriquer et en échange tu m'achètes mon produit B.

Il existe aussi des jeux à somme constante : par exemple deux joueurs qui doivent occuper un maximum de positions sur un plateau. Les jeux à somme constante se ramènent facilement à des jeux à somme nulle. Par exemple la matrice des gains du tireur face au gardien de but se transforme aisément en un jeu équivalent où la matrice des gains est à somme nulle.

		Joueur B		Joueur B	
		G	D	G	D
Joueur A	G	1 0	0 1	-1 1	1 -1
	D	0 1	1 0	1 -1	-1 1

2.5. Jeu symétrique/asymétrique

Dans un **jeu symétrique** les deux joueurs ont les mêmes possibilités au départ et les mêmes stratégies disponibles. Les échecs sont un exemple de jeu symétrique (à part le fait que les pions blancs commencent). Bien sûr au fil du jeu, cette symétrie peut être rompue, par exemple si je perds mes deux cavaliers et que je n'ai rien pris à mon adversaire alors j'ai moins de possibilités que lui.

Dans un **jeu asymétrique** les deux joueurs n'ont pas les mêmes ressources ou pas les mêmes capacités. Par exemple deux pays en guerre n'ont pas les mêmes ressources, ni territoires, ni armes. Autre exemple : dans un jeu vidéo chaque personnage a souvent ses propres caractéristiques (force/habileté/endurance/chance...).

2.6. Information complète/incomplète

Dans un jeu à **information complète** les joueurs connaissent toutes les règles et les gains possibles (par exemple les échecs). Dans un jeu à **information incomplète** les joueurs n'ont pas toutes les informations disponibles (comme dans la vraie vie économique ou les relations internationales). Par exemple les joueurs peuvent ne pas connaître quels sont les gains potentiels précis, quels sont les coups possibles, ni ce que l'adversaire a la possibilité de répliquer. Exemples : on peut ajouter un brouillard sur une carte d'un jeu vidéo afin de cacher les événements ou les ressources lointaines ; le fabricant d'avion sort un nouveau modèle d'avion sans savoir si les compagnies aériennes vont être intéressées par ce modèle ; on peut enchérir pour l'achat d'un tableau sans connaître sa valeur absolue.

2.7. Information parfaite/imparfaite

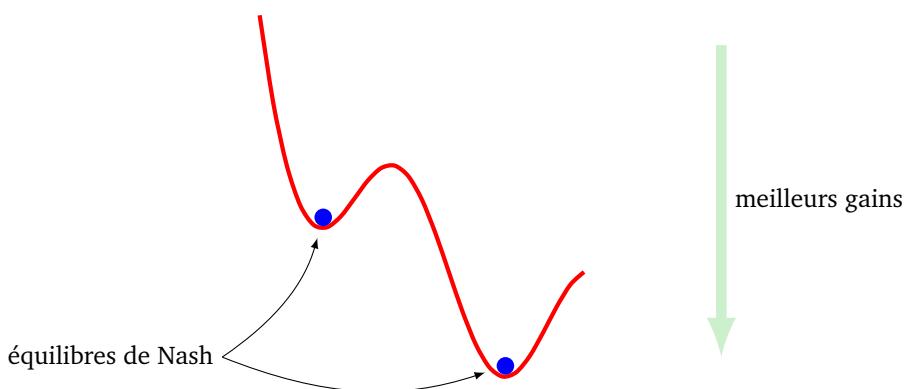
La notion d'**information parfaite** est différente : je sais exactement ce que peut jouer l'adversaire au coup suivant (même si je ne sais pas exactement ce qu'il va jouer), c'est le cas des échecs par exemple.

Voici des exemples où l'information est imparfaite : le fabricant d'avion A sort un nouveau modèle d'avion sans savoir si le concurrent B va lui aussi produire un modèle sur le même segment ; pour la plupart des jeux de cartes vous ne savez pas quelles sont les cartes de vos adversaires (belote, tarot, poker...).

3. Équilibre de Nash

3.1. Motivation

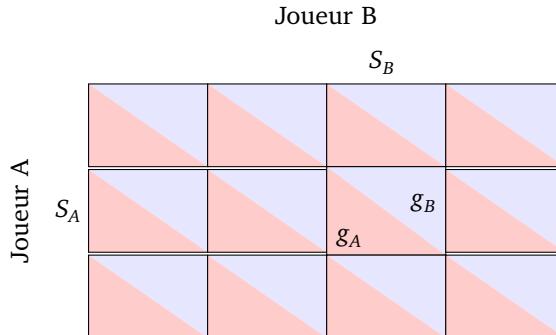
Dans un jeu, le but de chaque joueur est évidemment de gagner. Mais quelle stratégie permet d'avoir les meilleurs résultats ? Nous allons voir la notion « d'équilibre de Nash », mais attention ce n'est pas nécessairement la meilleure stratégie possible. C'est plutôt une position stable qui me dit que je n'ai pas intérêt à changer ma stratégie et mon adversaire non plus. L'analogie se fait avec une bille dans un creux qui, si on la déplace un peu, revient à sa position de départ.



Ce qu'un joueur espère trouver est une **stratégie dominante** qui est la stratégie (ou le prochain coup) qui est la meilleure possible quelle que soit l'action de l'adversaire. Par exemple, aux échecs, faire « échec et mat en trois coups » signifie qu'avec le coup que vous venez de jouer, quoi que fasse votre adversaire, vous êtes sûr de gagner lors des trois prochains coups. Bien évidemment une telle stratégie est très difficile, voire impossible à trouver, sinon c'est que le jeu n'est pas très intéressant. Un exemple classique est le jeu de Nim dans lequel tour à tour chaque joueur doit retirer 1, 2 ou 3 bâtons du tas initial; celui qui retire le dernier bâton a perdu. Pour celui qui commence il existe une stratégie qui lui permet de gagner à coup sûr.

3.2. Théorème de Nash

L'équilibre de Nash est l'ensemble des stratégies de deux joueurs considérées comme les meilleures possibles lorsque l'on fixe les choix de l'adversaire. Pour formaliser cette notion : notons S_A une stratégie adoptée par le joueur A et S_B une stratégie adoptée par le joueur B. La matrice des gains du jeu fournit la liste des gains : notons $g_A(S_A, S_B)$ le gain du joueur A s'il joue la stratégie S_A alors que B joue la stratégie S_B .



Définition.

Un couple (S_A, S_B) est un **équilibre de Nash** :

- si A n'a pas d'intérêt à changer sa stratégie S_A , sachant que B joue S_B , c'est-à-dire :

$$\text{pour toute stratégie } S \quad g_A(S_A, S_B) \geq g_A(S, S_B)$$

- et si B n'a pas d'intérêt à changer sa stratégie S_B , sachant que A joue S_A , c'est-à-dire :

$$\text{pour toute stratégie } S \quad g_B(S_A, S_B) \geq g_B(S_A, S)$$

En d'autres termes, lorsque les deux joueurs ont atteint un équilibre de Nash, ils n'ont pas intérêt à changer unilatéralement de stratégie.

Nous énonçons de manière informelle :

Théorème 1 (Théorème de Nash).

Un jeu admet toujours un (ou plusieurs) équilibre(s) de Nash.

Il y a une subtilité dans cette notion d'équilibre de Nash qui peut être **pur** (c'est-à-dire issu de la liste originale des stratégies possibles) ou bien **mixte** (les stratégies d'équilibres se forment par tirage aléatoire parmi les stratégies originales selon des probabilités précises).

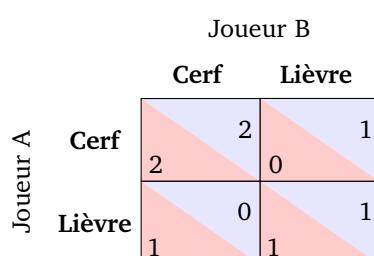
Trouver les équilibres de Nash purs.

Voici comment déterminer tous les équilibres de Nash purs à partir de la matrice de gains :

Un équilibre de Nash pur est atteint en une case où le gain de A est le plus grand de sa colonne et, en même temps, le gain de B est le plus grand de sa ligne.

3.3. Exemples

Le jeu de cerf.



- Le couple (Cerf, Cerf) est un équilibre de Nash. En effet, si B choisit la stratégie Cerf, alors :

$$g_A(\text{Cerf}, \text{Cerf}) = 2 \quad g_A(\text{Lièvre}, \text{Cerf}) = 1$$

donc une fois que B a choisi Cerf, A n'a pas intérêt à passer de Cerf à Lièvre.

Réciproquement, si A a choisi Cerf, alors B n'a pas non plus intérêt de passer de Cerf à Lièvre car :

$$g_B(\text{Cerf}, \text{Cerf}) = 2 \quad g_B(\text{Cerf}, \text{Lièvre}) = 1$$

- Le couple (Lièvre, Lièvre) est aussi un équilibre de Nash. En effet si B a choisi Lièvre alors A n'a pas intérêt de passer de Lièvre à Cerf, de même si A a choisi Lièvre alors B n'a pas intérêt de passer de Lièvre à Cerf.
- Les couples (Lièvre, Cerf) et (Cerf, Lièvre) ne sont pas des équilibres de Nash. Par exemple pour (Cerf, Lièvre) le gain pour A est 0, il a donc intérêt à changer sa stratégie (car (Lièvre, Lièvre) lui rapporterait 1).

Les deux équilibres de Nash (Cerf, Cerf) et (Lièvre, Lièvre) sont des équilibres de Nash purs. Il existe aussi un équilibre de Nash mixte (voir plus loin).

Le dilemme du prisonnier.

		Joueur B	
		S	T
Joueur A	S	-1	0
	T	-5	-3

- Le couple (T, T) est un équilibre de Nash. En effet pour A , (S, T) est pire et pour B , (T, S) est pire.
- Le couple (S, S) n'est pas un équilibre de Nash : en effet si B a choisi d'être solidaire alors A a tout intérêt à le trahir.
- Les couples (S, T) et (T, S) ne sont pas non plus des équilibres de Nash.

Dans cet exemple il n'y a qu'un équilibre de Nash, qui est pur.

Ainsi un équilibre de Nash n'est pas forcément la solution la meilleure d'un point de vue d'une personne extérieure, pour laquelle il est clair que le meilleur couple de stratégies pour l'ensemble des deux joueurs est (S, S) qui minimise le nombre total d'années de prison.

Le tir au but.

		Joueur B	
		G	D
Joueur A	G	1	0
	D	0	1

Il est facile de voir qu'il n'y a aucun équilibre de Nash pur. Par exemple la stratégie (G,G) n'est pas de Nash car le joueur A préférera (D,G) ; de même la stratégie (D,G) n'est pas de Nash car le joueur B préférera (D,D)...

Le théorème de Nash affirme pourtant qu'il existe un équilibre de Nash. Quel est-il ? Ici l'équilibre de Nash est un équilibre mixte. Pour chaque joueur sa stratégie d'équilibre est $\frac{1}{2}G + \frac{1}{2}S$, c'est-à-dire choisir G ou S avec chacun une probabilité $\frac{1}{2}$. Ainsi la meilleure stratégie pour le tireur (et aussi le gardien de but) c'est de décider au hasard gauche ou droite.

On se convainc facilement que ce sont les bonnes probabilités. Supposons par exemple que A ait pour stratégie $pG + (1-p)D$ où $0 \leq p \leq 1$ est une probabilité et supposons $p \geq \frac{1}{2}$ (c'est-à-dire que A tire plus souvent à gauche). Dans ce cas le gardien B a aussi intérêt à plonger lui aussi le plus souvent à gauche, donc il adopte une stratégie $qG + (1-q)D$ avec $q \geq \frac{1}{2}$. Mais puisque le gardien B part le plus souvent à gauche, le tireur A a intérêt à changer sa stratégie pour tirer le plus souvent à droite (donc prendre $p \leq \frac{1}{2}$), la seule solution est donc $p = \frac{1}{2}$ (et de même $q = \frac{1}{2}$).

Autre exemple d'équilibre mixte : pour « pierre/feuille/ciseaux » l'équilibre de Nash est atteint pour les deux joueurs avec la stratégie $\frac{1}{3}P + \frac{1}{3}F + \frac{1}{3}C$.

Exemple à trois stratégies.

Voici un exemple où le joueur A a trois stratégies possibles S_1, S_2, S_3 et le joueur B a trois autres stratégies possibles S'_1, S'_2, S'_3 . Noter que les stratégies n'ont pas besoin d'être les mêmes pour A et B et que la matrice des gains n'est pas symétrique.

		Joueur B		
		S'_1	S'_2	S'_3
Joueur A	S_1	3	1	0
	S_2	0	3	1
	S_3	4	2	1

Le couple de stratégies (S_3, S'_1) de gains $(4, 3)$ est un équilibre de Nash : la valeur 4 est la valeur la plus élevée du gain de A pour la première colonne et la valeur 3 est la valeur la plus élevée du gain de B pour la troisième ligne. De même, le couple de stratégies (S_2, S'_3) de gains $(2, 4)$ est un équilibre de Nash. Il existe aussi des algorithmes pour déterminer les équilibres de Nash mixtes.

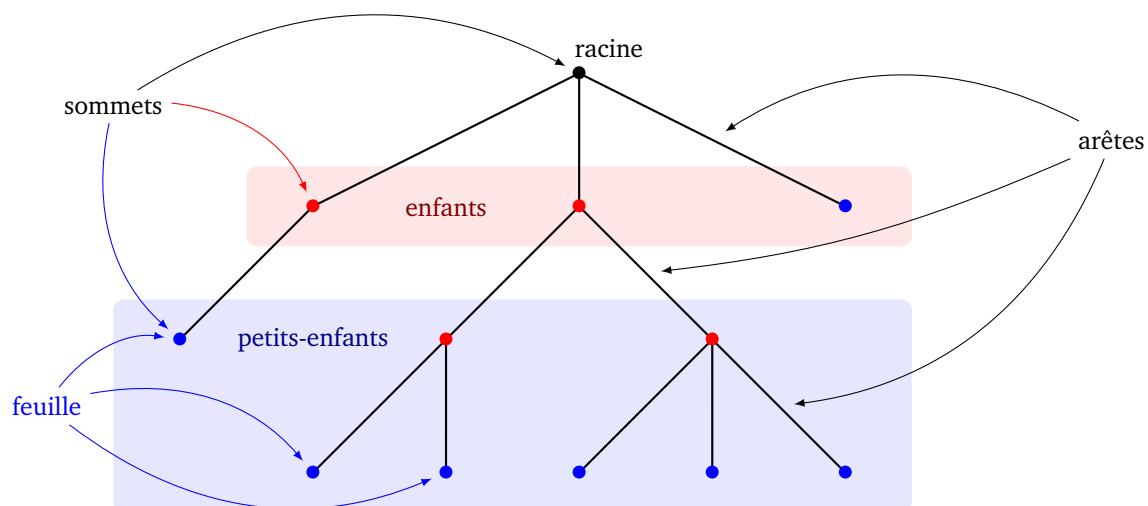
L'algorithme minimax permet de choisir le meilleur coup à jouer en anticipant les mouvements de l'adversaire.

1. Arbres

1.1. Vocabulaire

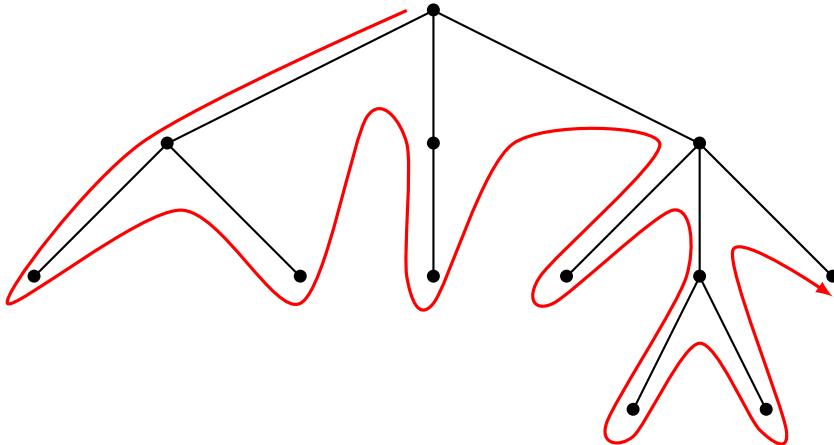
Voici le vocabulaire de base sur les arbres.

- Un **arbre** est un graphe sans cycle.
- Comme un graphe, un arbre est donc formé par des **sommets** (aussi appelés **nœuds**) reliés par des **arêtes**.
- La **racine** est un sommet privilégié qui sert d'origine. Cela définit une orientation naturelle sur l'arbre issue de la racine.
- Les sommets reliés à la racine sont les **enfants**; les sommets reliés aux enfants (autres que la racine) sont les **petits-enfants**...
- Plus généralement les **enfants d'un sommet** s sont les sommets s' reliés à s avec s' plus éloigné de la racine que s .
- Une **feuille** (ou **sommet terminal**) est un sommet sans enfant.
- Une **donnée** sera généralement associée à chaque sommet (un nombre, une configuration...).



1.2. Parcours en profondeur

Nous souhaitons parcourir tous les sommets d'un arbre. Le **parcours en profondeur** permet d'atteindre tous les sommets en partant de la racine puis en balayant l'arbre de haut (la racine) en bas (les feuilles) et de la gauche vers la droite.



L'algorithme est assez court mais comme c'est un algorithme récursif c'est toujours un peu délicat à comprendre. L'algorithme est donné sous la forme d'une fonction récursive `parcours_profondeur(sommet)`. Pour parcourir tout l'arbre on l'appelle par `parcours_profondeur(racine)`. Dans cet algorithme, on peut effectuer une action sur le sommet à l'aide d'une fonction `traiter(sommet)` à personnaliser : par exemple on affiche la donnée attachée à ce sommet.

Algorithme.

Fonction : `parcours_profondeur(sommet)`

Entrée : un sommet d'un arbre.

Action : traite tous les descendants de ce sommet par un parcours en profondeur.

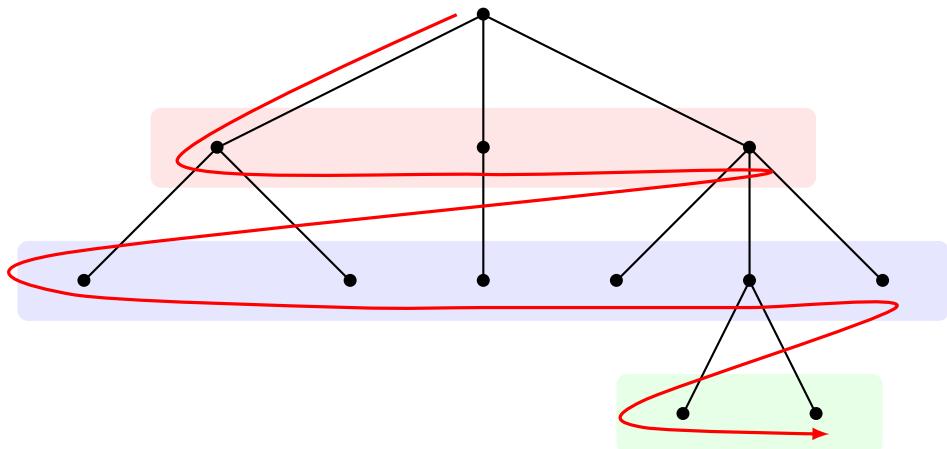
- `traiter(sommet)`
- pour chaque 'enfant' de sommet :
 - `parcours_profondeur(enfant)`

L'idée est la suivante : on part de la racine, on effectue l'action voulue sur ce sommet par une fonction `traiter(sommet)`, puis on considère l'ensemble des enfants de la racine. On considère le premier enfant (celui le plus à gauche), l'appel récursif fait que l'on traite ce sommet, puis on considère ses enfants de la gauche vers la droite (c'est-à-dire les petits-enfants de la racine) et on descend ainsi dans l'arbre. Une fois qu'on a traité tous les descendants du premier enfant, on passe au second enfant de la racine... Comme l'algorithme est récursif il est important de bien comprendre la condition d'arrêt : ici tout simplement si un sommet n'a pas d'enfant (c'est une feuille) alors on effectue juste l'action sur ce sommet (sans appel récursif) car la boucle « pour chaque 'enfant' » est vide.

Le mieux est de programmer soi-même cette fonction pour en comprendre vraiment son mécanisme, voir la section « Avec Python » un peu plus loin.

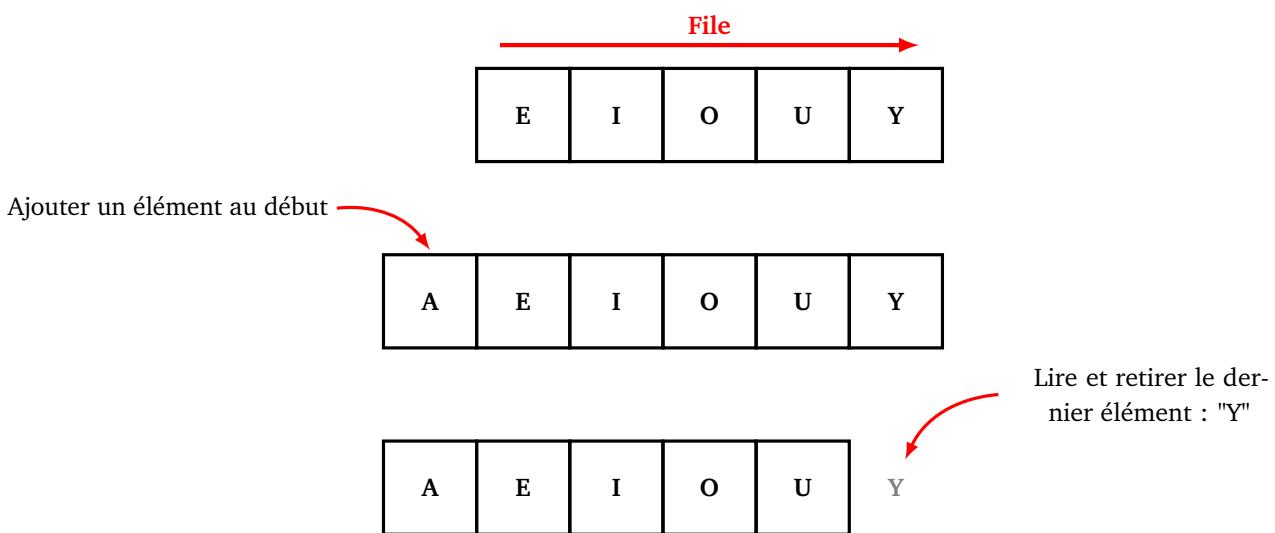
1.3. Parcours en largeur

C'est la façon qui semble humainement naturelle de parcourir un arbre mais un peu plus difficile pour un ordinateur. Le but est de partir de la racine, puis de traiter tous les enfants (de gauche à droite), puis l'ensemble de tous les petits-enfants, puis l'ensemble des petits-petits-enfants...



Avant d'étudier l'algorithme il faut comprendre la notion de « file ». Une **file** (*queue*) est une suite de données à laquelle on peut ajouter des éléments d'un côté et en retirer de l'autre. C'est comme une file d'attente : le premier arrivé sera le premier servi (*fifo* : *first in, first out*). Plus précisément voici les trois opérations de base :

- `file_vide()` : est-ce que la file est vide ?
- `ajouter_file(element)` : ajoute un élément à la gauche de la file,
- `retirer_file()` : renvoie l'élément le plus à droite de la file, et le retire de la file.



Il ne faut pas confondre une file avec une « pile » (*stack*) où le dernier élément ajouté à la pile est le premier à être retiré (*filo* : *first in, last out*).

Voici l'algorithme de parcours en largeur. Dans cet algorithme la file sert de stockage pour mémoriser les enfants d'un sommet, car il faut auparavant passer aux frères et sœurs de ce sommet.

Algorithme.

Fonction : `parcours_largeur(racine)`

Entrée : un sommet (la racine de l'arbre).

Action : traite tous les sommets par un parcours en largeur.

- Partir d'une file qui contient uniquement le sommet `racine`.
- Tant que la file n'est pas vide :
 - `sommet = retirer_file()`

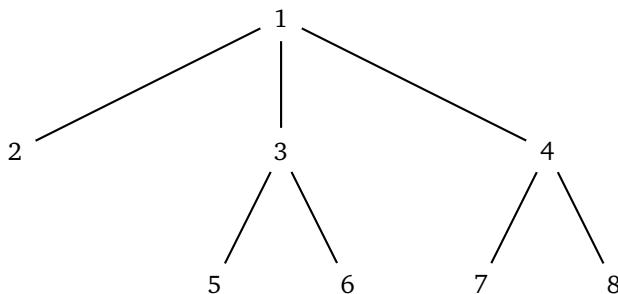
- traiter(sommet)
- pour chaque 'enfant' de sommet :
 - ajouter_file(enfant)

1.4. Avec Python

On commence par définir une classe d'objet `Noeud` qui correspond à un sommet avec une valeur et une liste d'enfants (qui sont eux-mêmes des `Noeud`).

```
class Noeud:
    def __init__(self, val):
        self.enfants = []
        self.val = val
```

Voici comment initialiser l'arbre suivant :



```
racine = Noeud(1)
enfant1 = Noeud(2)
enfant2 = Noeud(3)
enfant3 = Noeud(4)
racine.enfants = [enfant1, enfant2, enfant3]
petitenfant1 = Noeud(5)
petitenfant2 = Noeud(6)
enfant2.enfants = [petitenfant1, petitenfant2]
petitenfant3 = Noeud(7)
petitenfant4 = Noeud(8)
enfant3.enfants = [petitenfant3, petitenfant4]
```

Voici la fonction récursive qui permet un parcours d'arbre en profondeur (ici elle affiche juste la valeur associée à chaque sommet) :

```
def parcours_arbre_profondeur(noeud):
    print(noeud.val)
    for enfant in noeud.enfants:
        parcours_arbre_profondeur(enfant)
```

Sur l'arbre de notre exemple, cela affiche la valeur de chaque sommet :

1 2 3 5 6 4 7 8

Voici la fonction qui permet un parcours d'arbre en largeur :

```
def parcours_arbre_largeur(noeud):
    file = [noeud]
```

```

while file != []:
    noeud = file.pop()
    print(noeud.val)
    for enfant in noeud.enfants:
        file = [enfant] + file

```

Rappelons que la commande `file.pop()` correspond à `retirer_file()` et renvoie le dernier élément de la liste et le retire de cette liste.

Toujours pour le même exemple, voici maintenant l'ordre d'affichage (racine, puis enfants, puis petits-enfants) :

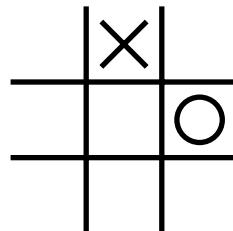
1 2 3 4 5 6 7 8

2. Algorithme minimax

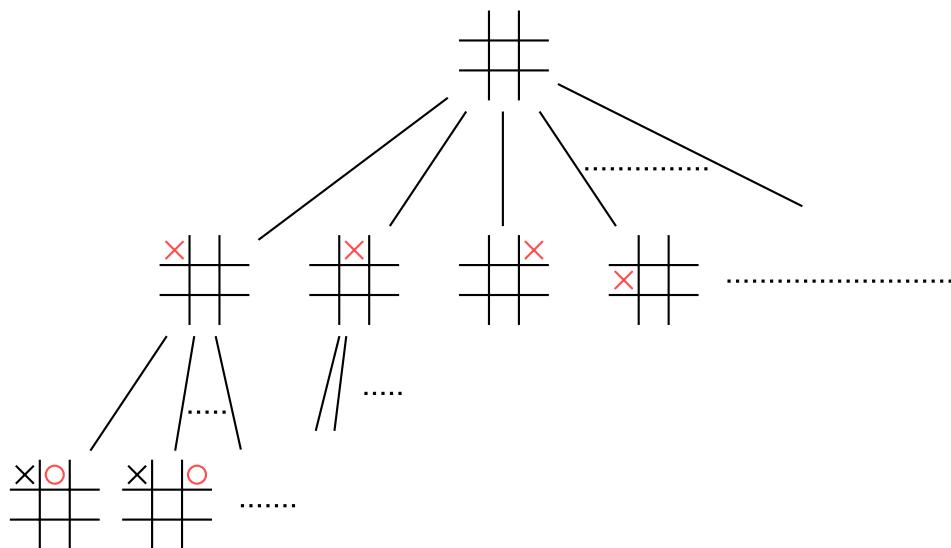
2.1. Jeux

Morpion.

Voyons comment les différentes successions de coups possibles d'une partie peuvent être codées sous la forme d'un arbre. Prenons l'exemple du jeu de morpion : sur une grille de 9 cases, il s'agit pour un joueur d'aligner 3 croix et pour son adversaire d'aligner 3 ronds.



La racine de l'arbre correspond à la position de départ (grille vide), la racine a 9 enfants qui correspondent aux 9 possibilités du premier joueur pour placer sa première croix. Chacun de ces enfants a lui-même 8 enfants qui correspondent aux 8 possibilités (les 8 cases vides restantes) pour le second joueur de placer son premier rond...

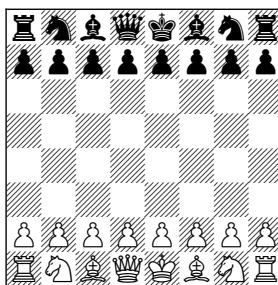


Les feuilles de l'arbre (les sommets sans enfants) correspondent soit à une configuration où l'un des joueurs a gagné, soit à une grille remplie sans vainqueur.

Dans une partie il y a au plus 9 coups. Le nombre total de parties est donc inférieur à $9! = 362\,880$ (9 choix pour le premier coup, 8 pour le second, 7 pour le troisième...). Si on tenait compte des symétries, alors il y aurait beaucoup moins de configurations possibles. Ainsi le morpion est un jeu aux possibilités plutôt limitées (mais vous le saviez déjà).

Échecs.

L'arbre des configurations pour décrire toutes les parties d'échecs possibles est immense ! Chaque joueur a 16 pièces, et il y a 64 cases, donc (en théorie) il y a jusqu'à $16 \times 64 = 1024$ possibilités, donc chaque sommet pourrait avoir plus de mille enfants. Dans la pratique on estime qu'un joueur a en moyenne 30 coups possibles et qu'une partie dure 80 coups. Cela fait un total de $30 \times 30 \times \dots = 30^{80} \simeq 10^{120}$ parties possibles. Ce nombre, 10^{120} , s'appelle le nombre de Shannon, il est gigantesque, pensez qu'il y a « seulement » 10^{80} atomes dans l'univers ! Ainsi il est impossible de calculer toutes les parties d'échecs possibles.



Ce que l'on peut faire avec un ordinateur, c'est chercher toutes les possibilités pour le prochain coup (disons 30 choix), puis pour chacun de ces coups chercher toutes les possibilités de l'adversaire (30 coups pour chacune des 30 possibilités, soit $30 \times 30 = 900$). Si on veut anticiper sur une profondeur de n coups, il faut calculer 30^n possibilités. Par exemple pour $n = 5$ cela fait environ 25 millions de parties à calculer ce qui est accessible. Par contre, pour $n = 7$ cela fait plus de 10 milliards de possibilités à calculer, ce qui n'est plus raisonnable.

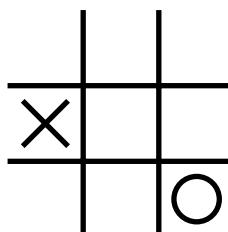
Nous allons faire un double travail :

- comment, parmi toutes ces possibilités, sélectionner la suite des coups les plus avantageux (tout en déjouant les stratégies de l'adversaire) ?
- ne pourrait-on pas accélérer le processus en écartant les configurations les plus défavorables ?

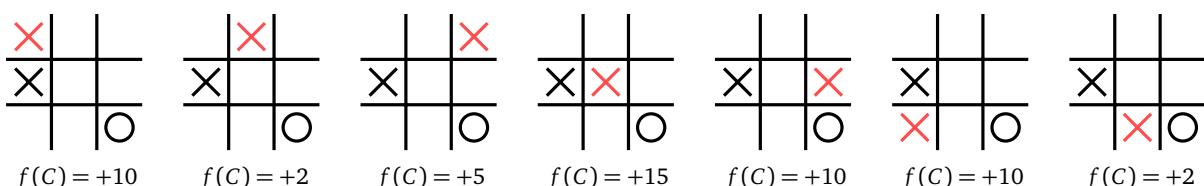
2.2. Fonction d'évaluation

Morpion.

Pour pouvoir décider du meilleur coup il faut pouvoir évaluer la qualité de chaque configuration de jeu. Reprenons l'exemple du morpion, je suis dans la situation suivante :

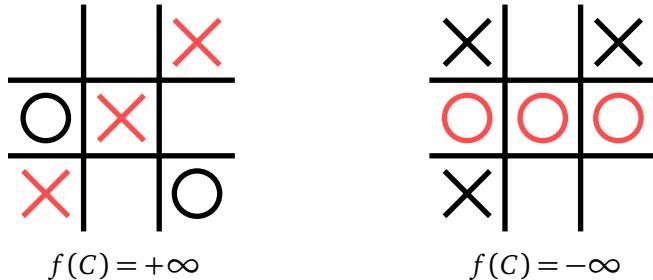


C'est à mon tour de placer une croix, j'ai donc 7 possibilités :



On va attribuer (ici un peu arbitrairement) une note $f(C)$ à chacune de ces configurations, une valeur haute désignant une configuration qui m'est favorable et me place en position de gagner la partie. Je veux éviter les valeurs basses (généralement négatives) qui sont des configurations favorables à mon adversaire.

Ainsi une **fonction d'évaluation** est une fonction f qui à chaque configuration de jeu associe un nombre réel. Plus ce nombre est grand, plus la configuration m'est favorable. Pour une configuration dans laquelle la partie est terminée et que j'ai gagné on peut attribuer la valeur $+\infty$ (ou bien un réel très grand). De même $-\infty$ (ou un réel très négatif) désigne une configuration où c'est mon adversaire qui a gagné.



Échecs.

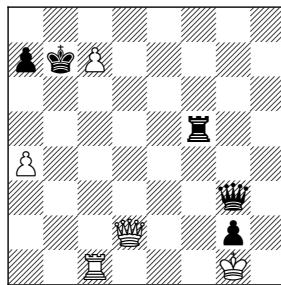
Créer une fonction d'évaluation pertinente est difficile et nécessite une bonne connaissance du jeu. Par exemple aux échecs on attribue souvent une valeur spécifique à chaque pièce comme ceci :

roi	reine	tour	fou	cavalier	pion
∞	10	5	3	3	1

La valeur du roi est infinie car cette pièce ne peut pas être capturée, elle est donc exclue de la fonction d'évaluation. Une fonction d'évaluation basique d'une configuration C est donc :

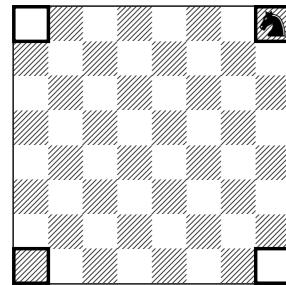
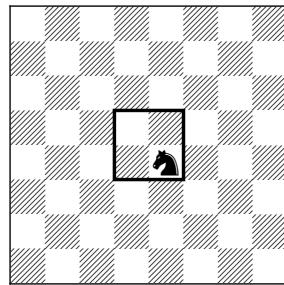
$$f(C) = \text{somme des valeurs des pièces blanches} - \text{somme des valeurs des pièces noires}$$

Par exemple, pour la configuration suivante la valeur des pièces blanches est $10 + 5 + 1 + 1 = 17$, la valeur des pièces noires est aussi 17, la fonction d'évaluation de cette configuration est donc $f(C) = 17 - 17 = 0$. Ce qui signifie que la situation est équilibrée. (D'ailleurs le prochain joueur qui joue gagne.)



$$f(C) = 0$$

Mais pour être plus pertinent, il faudrait aussi tenir compte de la position des pièces : par exemple un cavalier sur l'une des 4 cases centrales est dangereux, alors que placé dans un coin il ne sert à rien.



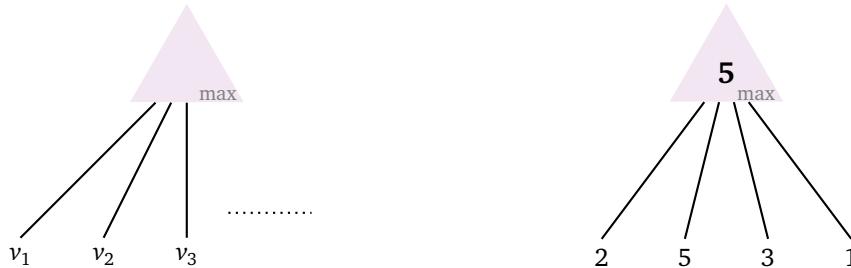
Noter que la fonction d'évaluation est commune aux deux joueurs, sauf que j'ai intérêt à trouver une configuration avec la valeur la plus grande possible alors que mon adversaire veut une configuration avec la valeur la plus petite possible.

Maximum et minimum.

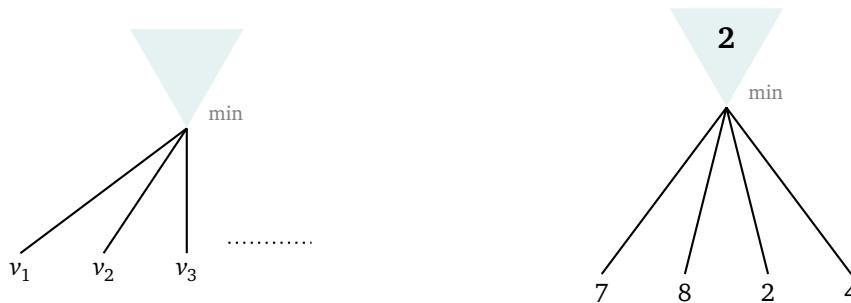
Si c'est à mon tour de jouer et si je ne veux (ou peux) que prévoir un coup à l'avance, alors :

- pour chaque coup numéro i possible, je calcule la configuration C_i obtenue,
- je lui attribue une valeur v_i , en calculant $v_i = f(C_i)$ par la fonction d'évaluation,
- je décide de jouer le coup qui réalise la plus grande valeur des v_i , c'est-à-dire je choisis i_{\max} tel que $v_{i_{\max}} = \max_i(v_i)$.

Voici comment on note cette situation en termes d'arbre. La racine sera dessinée sous la forme d'un triangle pointe vers le haut (pour signifier que le joueur cherche à maximiser), les enfants désignent toutes les configurations possibles pour mon prochain coup, ils sont représentés par la valeur v_i issue de la fonction d'évaluation.



Évidemment si c'est à mon adversaire de jouer, il calcule aussi tous ses coups possibles mais cette fois il cherche la configuration ayant la fonction d'évaluation la plus petite possible. Il choisit i_{\min} tel que $v_{i_{\min}} = \min_i(v_i)$. On représente cette fois la racine de l'arbre avec un triangle pointe en bas.



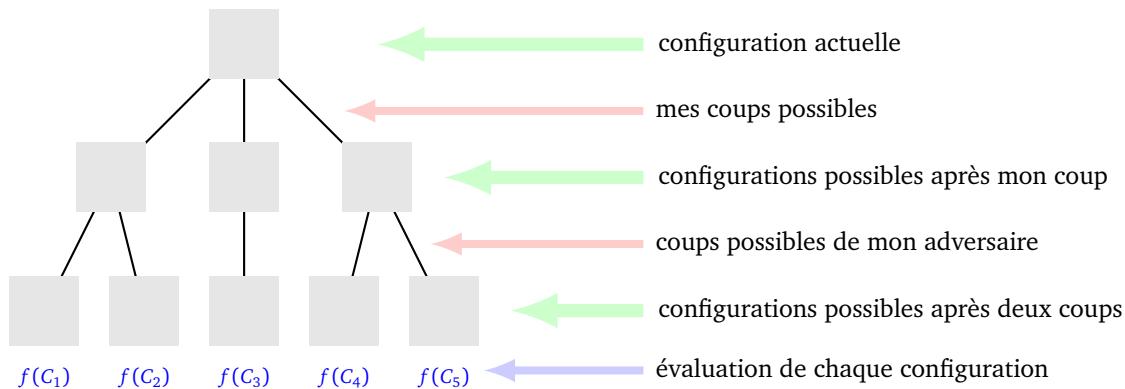
2.3. Algorithme minimax

Avant de détailler l'algorithme donnons les explications générales.

Construction de l'arbre

On va décider du meilleur coup à jouer en anticipant sur les n prochains coups. Pour cela on commence par construire l'arbre de toutes les configurations possibles :

- la racine correspond à la configuration actuelle,
- les enfants correspondent aux coups que je peux jouer ; plus précisément chaque enfant est la configuration obtenue après un des mes coups possibles,
- les petits-enfants correspondent aux configurations après un de mes coups suivi d'un coup de mon adversaire,
- etc.

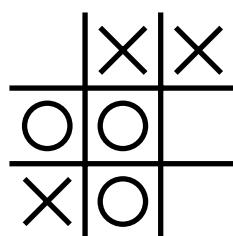


Évaluation sur les feuilles

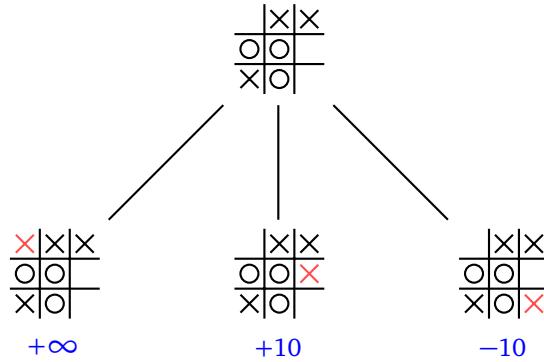
Les feuilles de notre arbre correspondent aux différentes configurations possibles dans n coups. Nous attribuons une valeur à chacune de ces configurations à l'aide de la fonction d'évaluation. (Nous évaluons seulement les configurations sur les feuilles, il est inutile d'évaluer les configurations intermédiaires.) Bien sûr, parmi toutes ces configurations j'aimerais arriver à la position ayant la valeur maximale mais mon adversaire fait tout ce qu'il peut pour m'en empêcher.

Exemple.

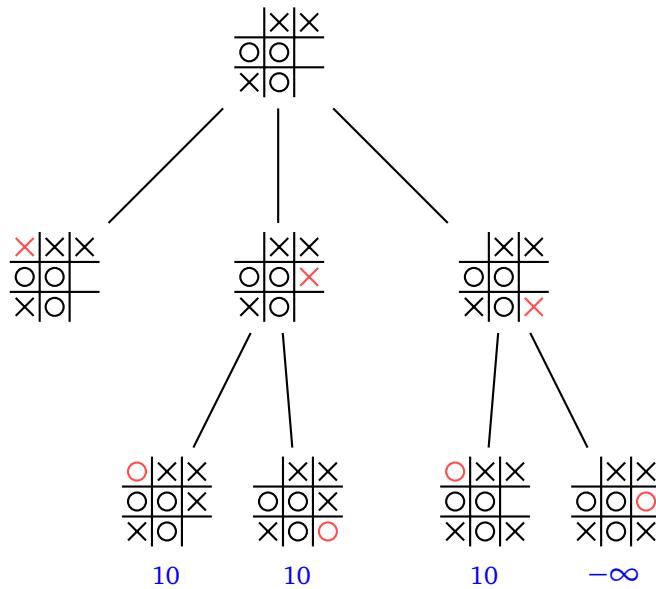
Je joue au morpion, voici la situation actuelle.



C'est à mon tour de jouer. Voici la liste des coups possibles. Pour chaque nouvelle situation possible, j'attribue une évaluation (ici donnée arbitrairement à titre d'exemple). L'évaluation est haute lorsque la configuration m'est favorable et l'évaluation est basse (et même négative) lorsque la situation est favorable à mon adversaire. L'évaluation $+\infty$ signifie une configuration gagnante (la partie est terminée, j'ai gagné) et $-\infty$ une situation perdante (la partie est terminée, j'ai perdu).



Voici maintenant tous les coups possibles de mon adversaire, ainsi que l'évaluation.



Appliquer l'algorithme minimax

Voici le déroulement de l'algorithme minimax expliqué avec des mots.

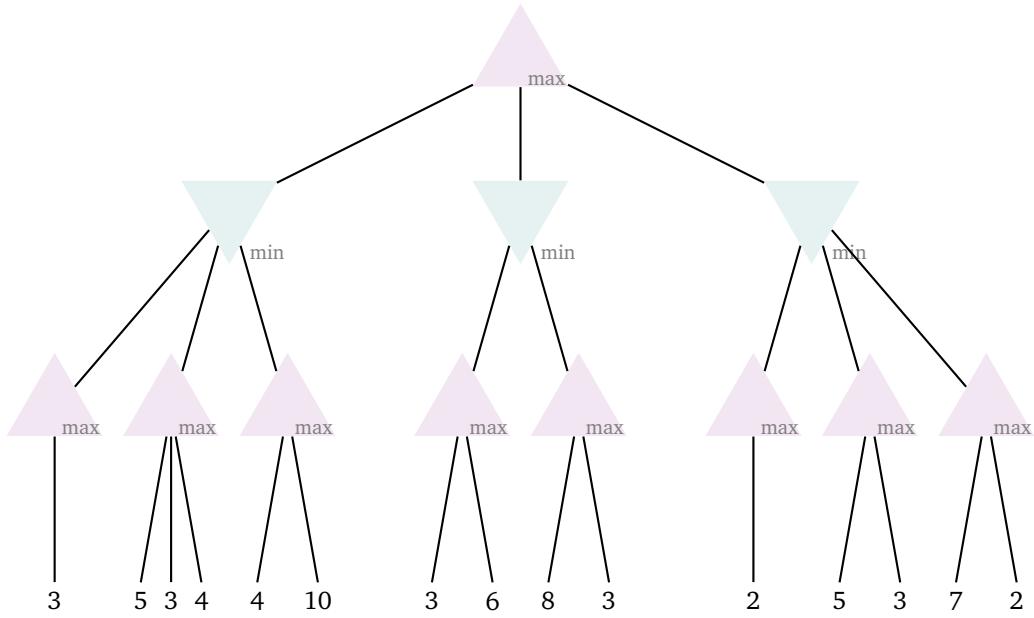
- On part des feuilles (tout en bas de l'arbre), à chaque feuille est associée une valeur obtenue par la fonction d'évaluation. On va remplir l'arbre avec des valeurs, du bas vers le haut.
- Pour chaque sommet de type « max » (avec un triangle vers le haut, qui correspond à mon tour de jouer) la valeur associée est le maximum de la valeur de ses enfants.
- Pour chaque sommet de type « min » (avec un triangle vers le bas, qui correspond à un coup à jouer par l'adversaire) la valeur associée est le minimum de la valeur de ses enfants.
- Partant des feuilles on associe, en remontant du bas vers le haut, une valeur à chaque sommet en terminant par la racine. On mémorise le chemin de la racine vers la feuille qui a permis de réaliser la valeur maximale : cela donne le prochain coup à jouer (ainsi que les coups suivants).

Noter qu'en général je ne peux pas espérer obtenir le maximum possible des valeurs aux feuilles, car mon adversaire va tout faire pour m'en empêcher. Pour cet algorithme, je suppose que mon adversaire fait la

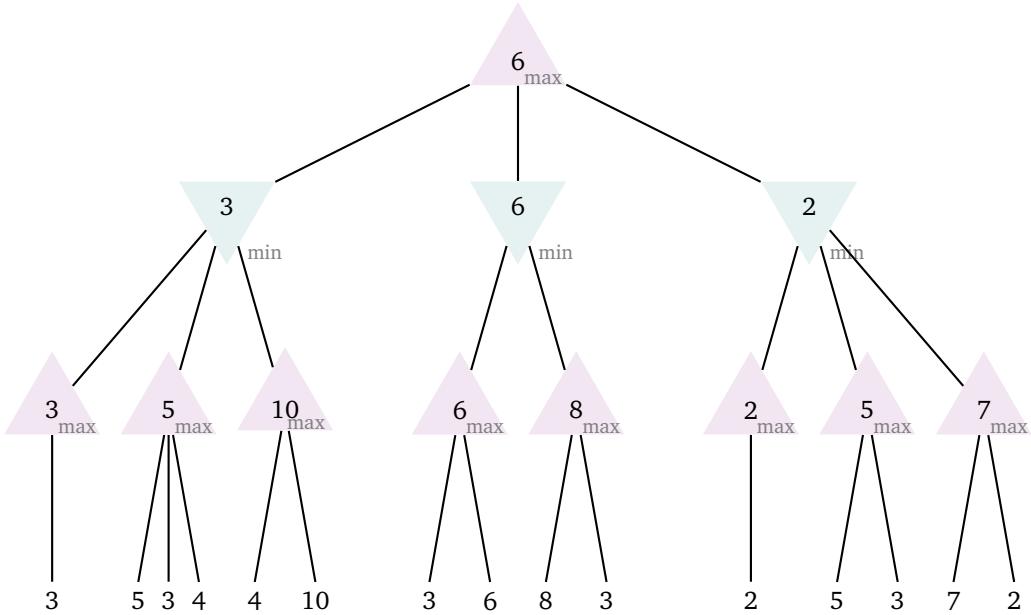
même évaluation que moi, mais sans prévoir plus de coups que moi.

Exemples

Voici l'arbre à remplir, seules les évaluations des positions dans trois coups sont notées.



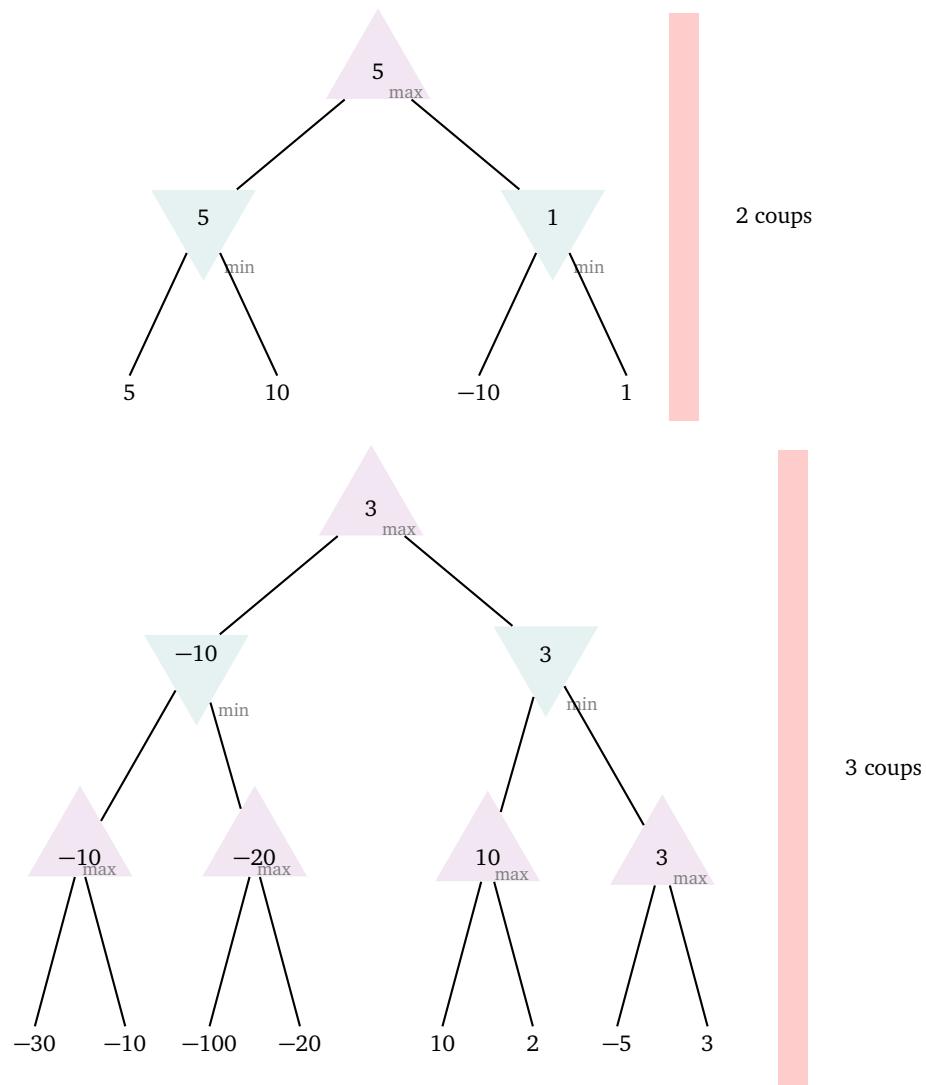
On remplit cet arbre du bas vers le haut, d'abord la ligne des maximums juste au-dessus des feuilles, puis la ligne des minimums et on termine par le maximum tout en haut qui nous donne la meilleure valeur que je puisse espérer.



Ainsi en prévoyant 3 coups à l'avance, je peut atteindre une position de valeur 6. Je ne peux pas espérer atteindre la meilleure position (de valeur 10) sauf si mon adversaire joue mal.

Horizon et sacrifice de la reine. Bien sûr le fait de ne pouvoir prévoir qu'un nombre limité de coups à l'avance ne garantit pas la victoire. Soit l'adversaire anticipe davantage de coups que nous, soit une stratégie qui s'avère bonne à court terme est en fait désastreuse à long terme.

Voici un exemple, dans lequel (arbre du haut) je prévois 2 coups à l'avance et pour lequel je vais choisir la branche de gauche qui semble m'apporter une position favorable (par exemple je prends la reine de mon adversaire), mais si j'avais pu prévoir 3 coups à l'avance (arbre du bas) alors je me serai aperçu que c'était un piège et que la branche de gauche me place dans une position très défavorable.



L'algorithme en détail

Pour l'algorithme minimax nous aurons besoin de deux fonctions :

- la fonction d'évaluation, `evaluation(jeu)`, renvoie une valeur pour une configuration jeu. Cette fonction sert à évaluer les configurations sur les feuilles, ce sera notre étape terminale de la fonction récursive.
- Une fonction `joue_un_coup(jeu, coup)` qui renvoie le nouveau jeu correspondant au coup.

Algorithme.

Fonction : `minimax(jeu, profondeur, joueur_maximise)`

Entrée :

- `jeu` : une configuration, c'est-à-dire un sommet d'un arbre.
- `profondeur` : le nombre de coups à anticiper.
- `joueur_maximise` : vrai ou faux, selon que le joueur cherche à maximiser la fonction d'évaluation ou à la minimiser.

Sortie : le score maximal que je peux espérer (ou bien le score minimal que mon adversaire peut espérer).

- Si `profondeur = 0` alors renvoyer `evaluation(jeu)`.
- Si `joueur_maximise` est Vrai :
 - `score_max = -∞`
 - Pour chaque coup possible :
 - `nouveau_jeu = joue_un_coup(jeu, coup)`

```

    — score = minimax(nouveau_jeu, profondeur-1, Faux)
    — Si score > score_max, faire score_max = score.
    — Renvoyer score_max
• Si joueur_maximise est Faux :
    — score_min = +∞
    — Pour chaque coup possible :
        — nouveau_jeu = joue_un_coup(jeu, coup)
        — score = minimax(nouveau_jeu, profondeur-1, Vrai)
        — Si score < score_min, faire score_min = score.
    — Renvoyer score_min

```

L'appel initial se fait par `minimax(jeu, profondeur, Vrai)` à partir de la position jeu actuelle et du nombre de coups profondeur que je veux anticiper. On verra dans l'implémentation concrète comment renvoyer la liste des coups qui permettent de réaliser ce maximum (ou minimum).

2.4. Avec Python

Préliminaires :

- Une fonction `evaluation(jeu)` ; pour nos tests ci-dessous cette fonction renvoie une valeur aléatoire (entre 1 et 20).
- Une fonction `joue_un_coup(jeu, coup)` qui calcule et renvoie la nouvelle configuration du jeu après le coup. Pour nos tests elle renvoie le jeu sans modifications.

```

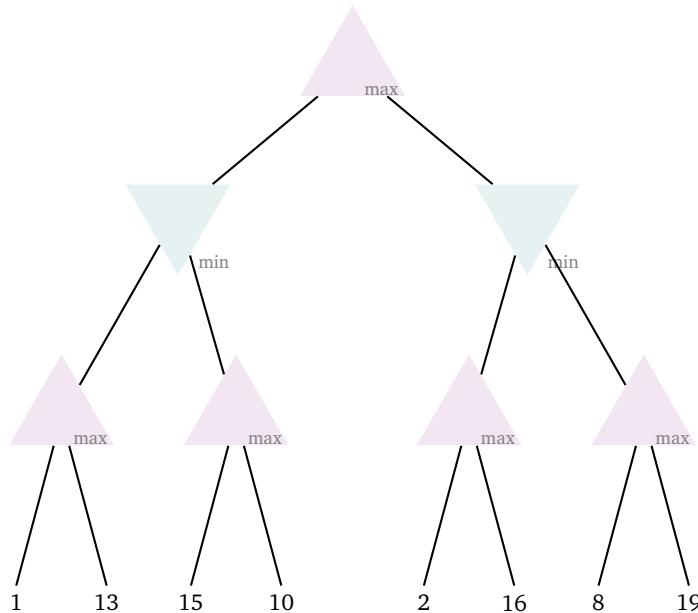
def minimax(jeu, prof, coups_prec, joueur_maximise):
    M = 3 # Ici M coups possibles à chaque fois
    if prof == 0:                      # On est à une feuille
        score = evaluation(jeu)
        print('Coups :',coups_prec, 'score :', score)
        return (score, coups_prec)
    else:
        if joueur_maximise:
            score_max = -math.inf
            for coup in range(M):
                nouv_jeu = joue_un_coup(jeu, coup)
                score, nouv_coups_prec =
                    minimax(nouv_jeu,prof-1,coups_prec+[coup],False)
                if score > score_max:
                    score_max = score
                    coups_max = nouv_coups_prec
            return (score_max, coups_max)
        else:
            score_min = +math.inf
            for coup in range(M):
                nouv_jeu = joue_un_coup(jeu, coup)
                score, nouv_coups_prec =
                    minimax(nouv_jeu,prof-1,coups_prec+[coup],True)
                if score < score_min:
                    score_min = score
                    coups_min = nouv_coups_prec

```

```
        return (score_min, coups_min)
```

Avec jeu = None (car on ne teste pas notre algorithme sur un vrai jeu), l'appel de la fonction est par exemple : minimax(jeu, 3, [], True) afin d'anticiper sur les 3 prochains coups. Ici la fonction renvoie le meilleur score attendu (comme l'algorithme vu précédemment) mais aussi la suite des coups qui permet d'atteindre ce maximum.

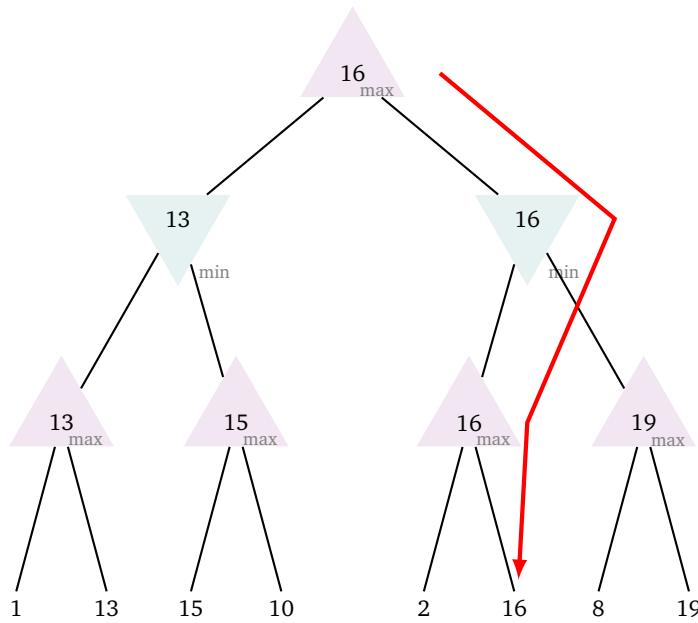
Voici un exemple correspondant à l'arbre :



Il y a deux enfants à chaque branche ($M = 2$) et le nombre de coups anticipés est 3.

```
Coups : [0, 0, 0] score : 1
Coups : [0, 0, 1] score : 13
Coups : [0, 1, 0] score : 15
Coups : [0, 1, 1] score : 10
Coups : [1, 0, 0] score : 2
Coups : [1, 0, 1] score : 16
Coups : [1, 1, 0] score : 8
Coups : [1, 1, 1] score : 19
```

```
Coups à jouer (16, [1, 0, 1])
```



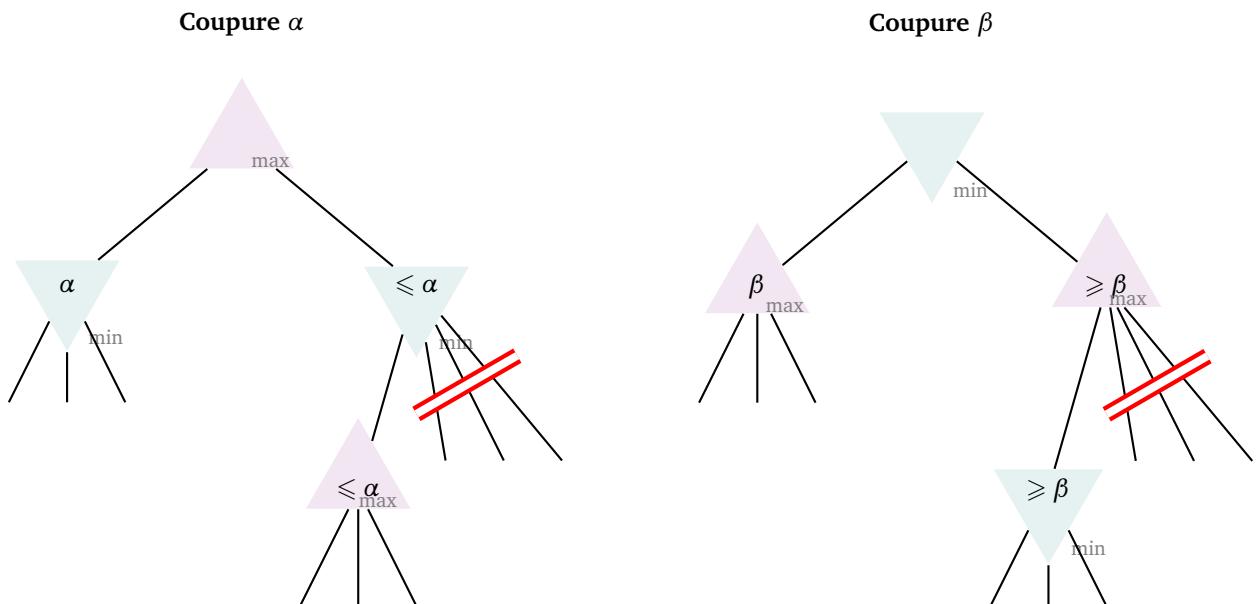
La valeur « minimax » est bien 16 et s'obtient par le chemin [1, 0, 1] c'est-à-dire branche de droite (1), puis branche de gauche (0) puis branche de droite (1 de nouveau).

3. Élagage alpha-beta

3.1. Idée

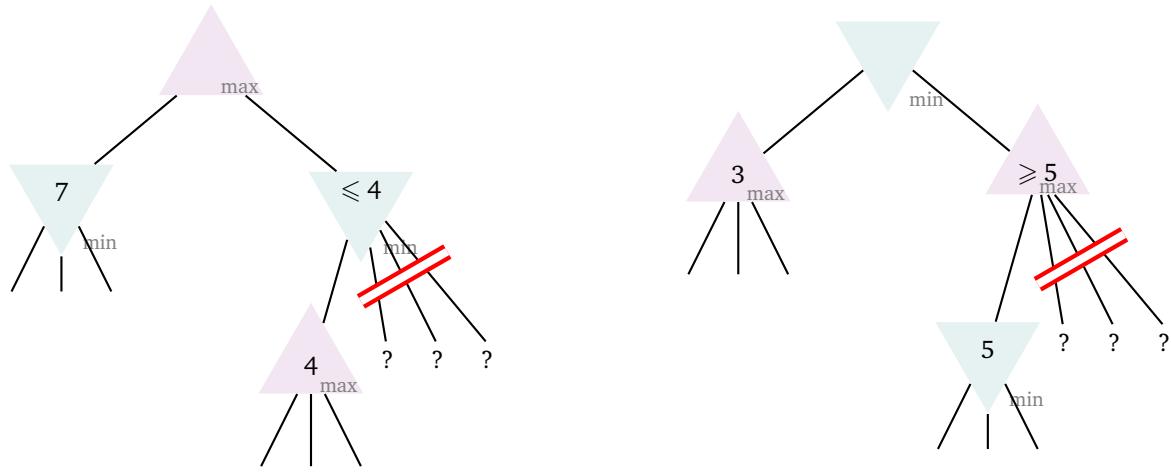
Rien ne sert de prévoir toutes les successions de coups lorsque de toute façon on a déjà trouvé mieux. Cela permet de réduire drastiquement le nombre de coups à évaluer. Si à chaque position il y a n possibilités et que l'on veut prévoir les d prochains coups, alors la complexité est en $O(n^d)$. La méthode dite « élagage alpha-beta » a pour complexité en moyenne $O(n^{\frac{d}{2}})$. Cela signifie qu'au lieu d'avoir à étudier n branches à chaque sommet, il y a seulement \sqrt{n} branches en moyenne à considérer. Par exemple si à chaque fois il y a 36 coups possibles, avec l'élagage alpha-beta il n'y a plus, en moyenne, que 6 branches à étudier.

3.2. Cas de base

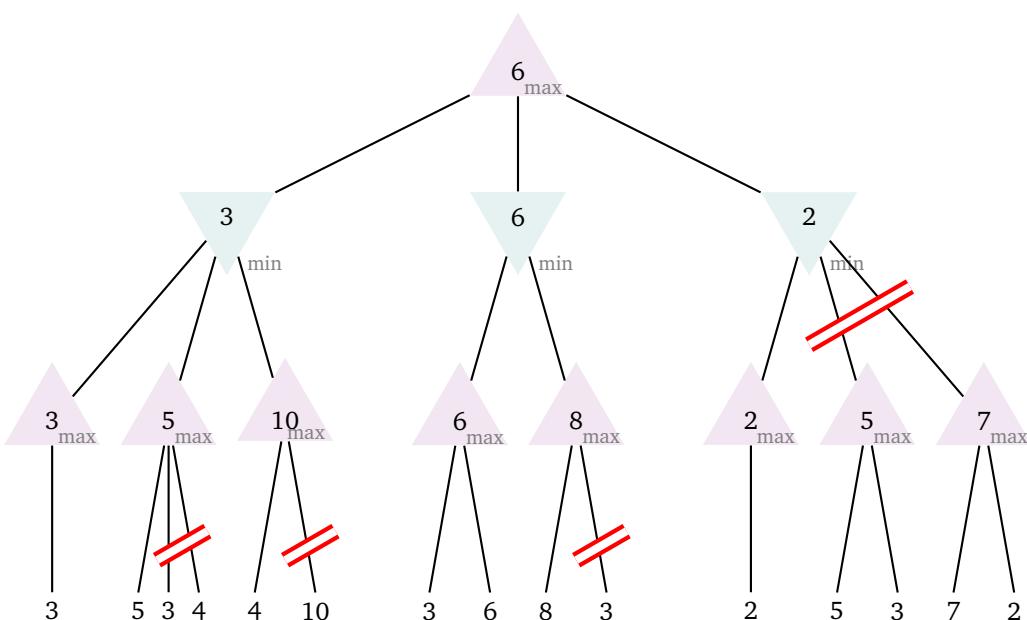


Exemple.

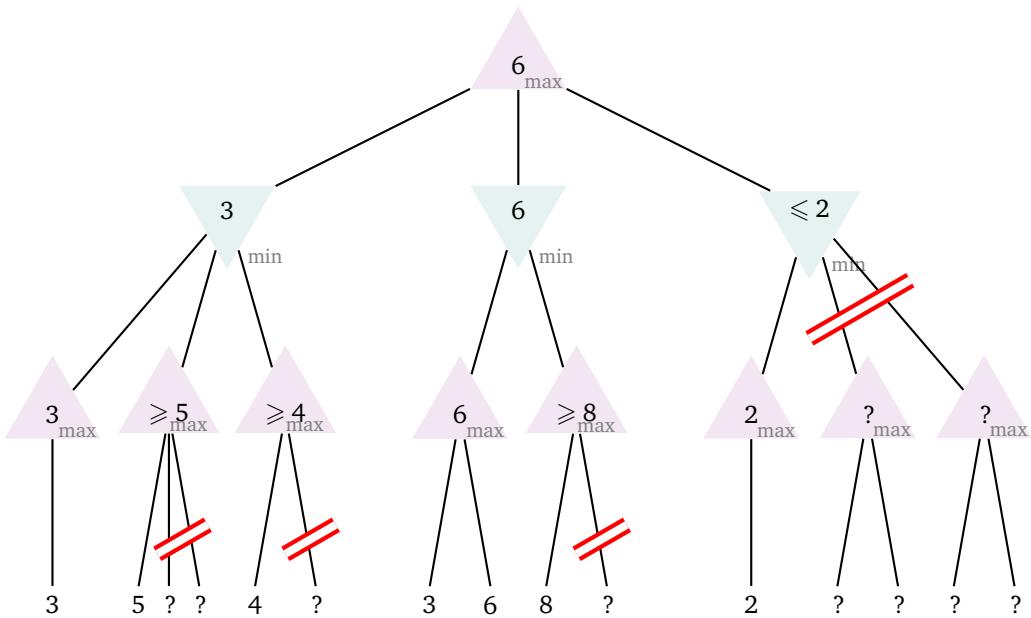
Sur les deux exemples ci-dessous, vous pouvez compléter la valeur finale dans le triangle du haut sans avoir besoin de connaître les valeurs (marquées par des « ? ») issues des branches coupées.

**3.3. Exemple**

Voici les branches que l'on pourrait élaguer pour l'exemple rencontré précédemment.



Cela signifie que l'on peut éviter 8 évaluations sur les 15. Ces évaluations non faites sont les feuilles marquées par des « ? » ci-dessous. Quelles que soient les valeurs qui remplacent ces « ? » le résultat final de l'algorithme (la valeur dans le triangle supérieur) est le même.



3.4. Algorithme

L'algorithme d'élagage est une adaptation de l'algorithme minimax. La fonction `alphabeta()` renvoie la même valeur que `minimax()` mais avec moins de calculs. α et β sont deux paramètres (on aura toujours $\alpha \leq \beta$). Lorsqu'on appelle la fonction la première fois on pose $\alpha = -\infty$ et $\beta = +\infty$ (ou bien des très grands nombres).

Algorithme.

Fonction : `alphabeta(jeu, profondeur, alpha, beta, joueur_maximise)`

- Si `profondeur = 0` alors renvoyer `evaluation(jeu)`.
- Si `joueur_maximise` est Vrai :
 - `score_max = -∞`
 - Pour chaque coup possible :
 - `nouveau_jeu = joue_un_coup(jeu, coup)`
 - `score = alphabeta(nouveau_jeu, profondeur-1, alpha, beta, Faux)`
 - Si `score > score_max`, faire `score_max = score`.
 - Si `score ≥ beta`, arrêter ici la boucle « pour ».
 - `alpha = max(alpha, score)`
 - Renvoyer `score_max`
- Si `joueur_maximise` est Faux :
 - `score_min = +∞`
 - Pour chaque coup possible :
 - `nouveau_jeu = joue_un_coup(jeu, coup)`
 - `score = alphabeta(nouveau_jeu, profondeur-1, alpha, beta, Vrai)`
 - Si `score < score_min`, faire `score_min = score`.
 - Si `score ≤ alpha`, arrêter ici la boucle « pour ».
 - `beta = min(beta, score)`
 - Renvoyer `score_min`

L'élagage α correspond à la première partie (lorsque le joueur chercher à maximiser le score), l'élagage β à la seconde partie (lorsque le joueur chercher à minimiser le score).

3.5. Avec Python

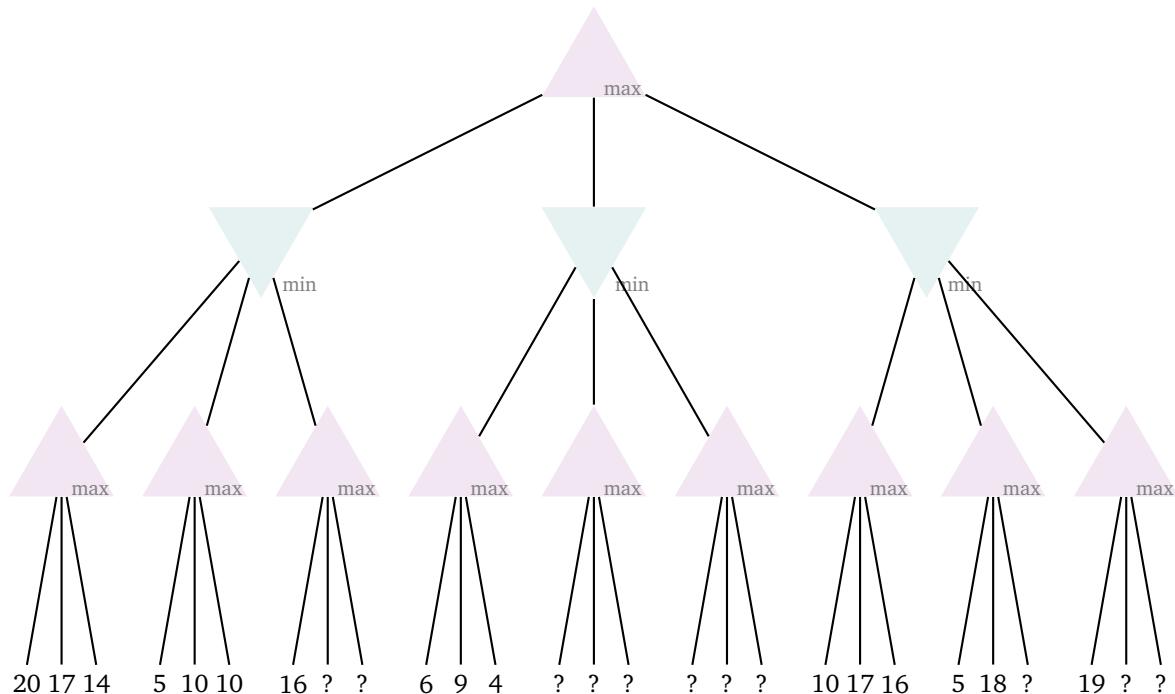
L'implémentation ne pose pas de difficulté particulière, il suffit de modifier le code précédent lorsque le joueur cherche à maximiser (faire une modification similaire pour minimiser) :

```

if score > score_max:
    score_max = score
    coups_max = nouv_coups_prec
if score >= beta: # Coupure alpha
    break           # On arrête de chercher les coups suivants

```

Voici un exemple avec une profondeur de 3 et $M = 3$ coups possibles à chaque fois. Sur les $3^3 = 27$ possibilités, l'élagage alpha-beta permet ici d'évaluer seulement 16 configurations.

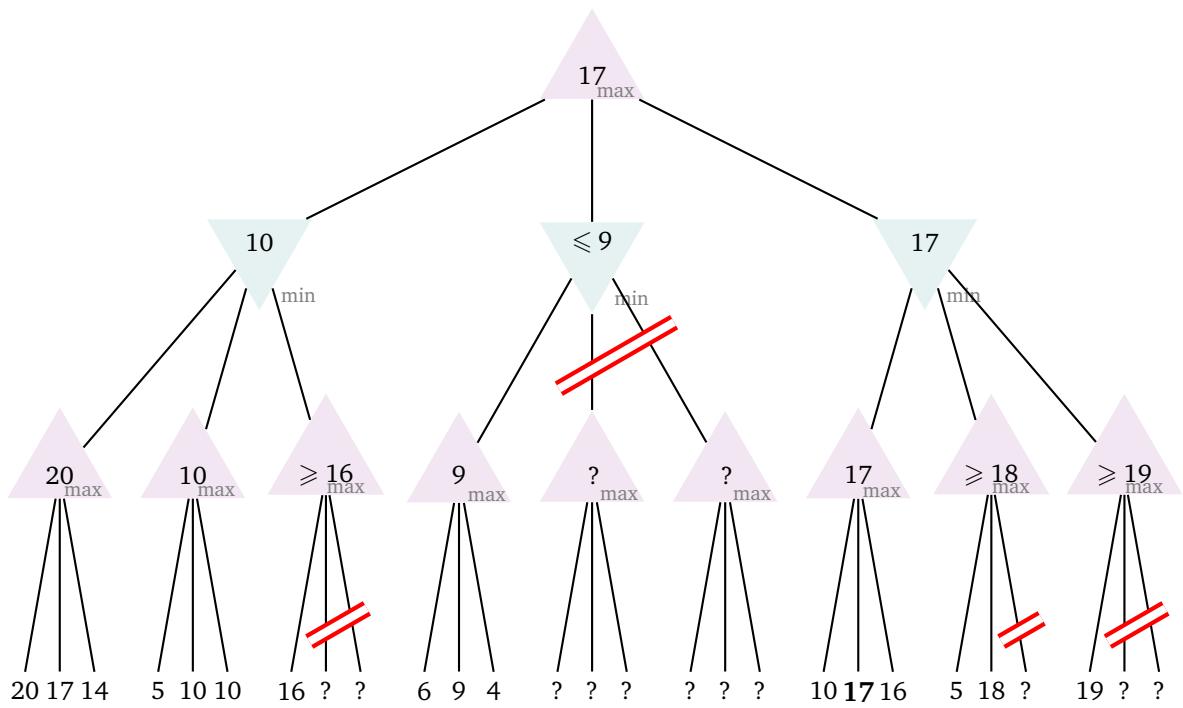


```

Coups : [0, 0, 0] score : 20
Coups : [0, 0, 1] score : 17
Coups : [0, 0, 2] score : 14
Coups : [0, 1, 0] score : 5
Coups : [0, 1, 1] score : 10
Coups : [0, 1, 2] score : 10
Coups : [0, 2, 0] score : 16
Coups : [1, 0, 0] score : 6
Coups : [1, 0, 1] score : 9
Coups : [1, 0, 2] score : 4
Coups : [2, 0, 0] score : 10
Coups : [2, 0, 1] score : 17
Coups : [2, 0, 2] score : 16
Coups : [2, 1, 0] score : 5
Coups : [2, 1, 1] score : 18
Coups : [2, 2, 0] score : 19

```

Coups à jouer (17, [2, 0, 1])



Sociologie du joueur

Quel est le comportement d'un joueur dans la « vraie vie » et quels sont les paramètres qui permettent de créer un « bon » jeu ?

1. Jeu réel

Aux échecs la meilleure stratégie est rationnelle. Il y a peu de chance que vous battiez un bon joueur en jouant certains coups au hasard ou selon votre humeur. Cependant, dans des jeux moins rationnels, le comportement humain n'est pas toujours celui que l'on attend. Le poker ou le black jack sont des exemples de jeux avec une forte part de hasard, mais il existe une stratégie rationnelle qui tient compte des probabilités en fonction des cartes découvertes et celles que vous avez en main. Cette stratégie ne vous garantit pas de gagner mais vous assure la meilleure *espérance* de gain, c'est-à-dire le meilleur gain possible si vous pouviez répéter l'expérience de nombreuses fois. Par exemple si je joue à pile ou face avec une pièce truquée qui tombe sur « pile » deux fois sur trois, alors la meilleure stratégie est bien évidemment de choisir « pile », cela ne m'assure pas de gagner à tous les coups, mais si je joue 1000 fois, je devrais gagner environ 666 fois. Nous allons voir que pour certains jeux le comportement humain n'est pas toujours le plus rationnel.

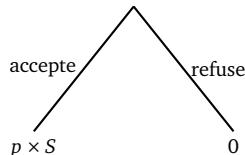
1.1. Jeu de l'ultimatum

Le jeu. Le *jeu de l'ultimatum* est un jeu à deux joueurs : le *proposant* et le *répondant*. On met à disposition du proposant une certaine somme d'argent (disons 100 euros), il doit faire une offre de partage à l'autre joueur (ex. « je te donne 20% de mes 100 euros »). Le répondant a deux possibilités :

- Il accepte l'offre (et alors le partage se fait selon la répartition convenue).
- Il refuse l'offre et alors *personne* ne gagne rien.

Le proposant et le répondant ne peuvent pas négocier, c'est une situation « à prendre ou à laisser » pour le répondant.

Voici l'arbre du gain du répondant, pour une somme initiale S et une proposition de pourcentage p .



Solution rationnelle. La seule stratégie rationnelle pour le répondant c'est d'accepter la proposition quelle soit ! En effet, en acceptant l'offre, il a un gain positif ou nul, toujours supérieur ou égal au gain nul s'il refuse. Sachant cela, la stratégie rationnelle du proposant serait d'offrir un pourcentage le plus petit possible (voire même 0).

Expérimentations. De nombreuses expériences de ce jeu ont été réalisées dans différents pays avec différentes cultures. Voici ce qui en ressort :

- Le proposant soumet généralement une part de la somme comprise entre 40% et 50%. Lorsque le répondant est de la même communauté que le proposant (famille, amis, même village) alors le pourcentage proposé est généralement 50%.
- Les offres de 40% et plus sont généralement acceptées par les répondants.
- Les offres de 20% et moins sont généralement refusées (avec une offre à 20%, environ 50% de rejet, avec une offre à 10% environ 90% de rejet).

C'est ce dernier point qu'il convient d'expliquer car du point de vue d'un bilan purement comptable, le répondant a tout simplement refusé de l'argent ! Les explications de ce refus avancées par les répondants sont :

- donner une leçon (au cas où la situation se répéterait dans le futur),
- punir (car le répondant trouve l'offre injuste et en souffre).

Cette expérience met en avant le besoin d'équité et de justice pour vivre en société.

Les expérimentations ont été menées avec des petites sommes (disons de 10 à 100 euros), pour des sommes plus élevées, on peut penser que les résultats seraient différents. Par exemple refuseriez-vous ma proposition de 10% de 1 000 000 d'euros ? Le jeu serait aussi différent s'il était répété plusieurs fois de suite (ou si on échangeait de rôle une fois sur deux). Voir le jeu *tit-for-tat* du chapitre « Théorie des jeux ».

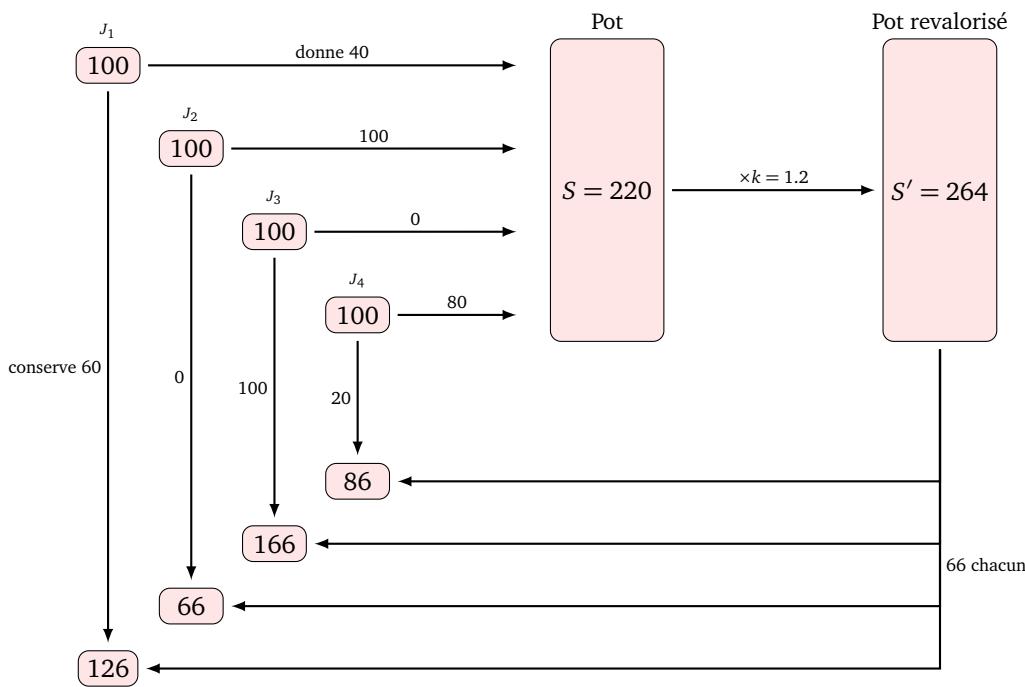
Dans le jeu de l'ultimatum, le répondant a intérêt à être égoïste et à garder le maximum d'argent pour lui en espérant que son offre soit acceptée. Dans le jeu suivant la situation est renversée : le joueur a intérêt à être le plus généreux possible.

1.2. Jeu à contribution

Le jeu. Voici comment jouer au *jeu à contribution* :

- il y a n joueurs,
- chaque joueur reçoit la même somme S ,
- chaque joueur décide secrètement de placer tout ou partie de sa somme dans le pot commun. Il garde pour lui le reste.
- La somme totale S du pot commun est multipliée par un facteur k (avec $1 < k < n$) : $S' = k \times S$.
- Le montant du pot commun revalorisé est réparti équitablement entre les n joueurs (S'/n).
- À la fin, chaque joueur dispose donc de l'argent qu'il n'a pas mis au pot, auquel s'ajoute le partage du pot commun revalorisé.

Exemple.



Prenons $n = 4$ joueurs qui disposent chacun de $S = 100$ avec un facteur multiplicatif qui est $k = 1.2$. Chacun décide de mettre respectivement 40, 100, 0, 80 dans le pot commun. Le total du pot est donc 220, qui une fois revalorisé est $k \times 220 = 264$. Le pot revalorisé est réparti en parts de 66 euros entre les quatre joueurs. À la fin :

- le joueur 1 a $60 + 66 = 126$ (les 60 qu'il n'a pas partagés et sa part du pot commun),
- le joueur 2 a $0 + 66 = 66$,
- le joueur 3 a $100 + 66 = 166$,
- le joueur 4 a $20 + 66 = 86$.

Meilleure stratégie. La meilleure stratégie collective c'est que tout le monde mette tout son argent au pot commun afin de profiter au maximum du facteur k . Dans l'exemple précédent les joueurs ont à la fin un total de 444 euros alors que s'ils avaient tout mis au pot commun ils se seraient partagés 480 euros (120 chacun).

Cependant ce qui peut arriver de mieux pour un joueur précis, c'est qu'il ne mette rien au pot commun mais que tous les autres donnent tout. Dans la situation précédente avec quatre joueurs, le joueur égoïste obtiendrait 190 euros et les autres 90 euros chacun.

L'impôt est une analogie de la vie courante : tout le monde paye des impôts et en échange profite des écoles, des routes... mais si une personne se débrouille pour ne pas en payer, elle profite à la fois de son argent pas dépensé et des infrastructures !

Expérimentations. Dans la pratique, les joueurs ont tendance à placer quelque chose dans le pot : une somme variant de 0 à 100% de la somme possible (mais rarement 0) et qui dépend aussi du facteur k . Les pourcentages se stabilisent si le jeu est répété, les joueurs ont tendance à baisser leur contribution lorsqu'ils s'aperçoivent que d'autres ont été moins généreux. Par exemple, avec un facteur $k = 1.3$, le pourcentage placé au pot commun est environ de 50%.

1.3. Partage de Shapley

Dans les jeux coopératifs (ou dans la vraie vie) chacun fournit une contribution plus ou moins importante. Si le groupe gagne une somme S , comment répartir les gains de façon équitable en tenant compte de la participation de chacun ? Une formule possible est donnée par le **partage de Shapley** du nom de Lloyd

Shapley (1923–2016), mathématicien qui a obtenu la récompense dite « prix Nobel d'économie » pour ses travaux.

Fonction valeur.

Le jeu se joue à n joueurs numérotés de 1 à n . On attribue un gain à chaque groupe de joueurs à l'aide d'une fonction v .

- On note $v(1), v(2), \dots, v(n)$ le gain obtenu par un joueur s'il jouait tout seul. Les joueurs ne sont pas nécessairement tous du même niveau, et donc les gains individuels peuvent être différents.
- On note $v(i, j)$ le gain de la coalition des deux joueurs i et j . Ainsi $v(1, 2)$ est le gain obtenu par le regroupement des joueurs 1 et 2.
- Plus généralement si E est sous-ensemble de $\{1, 2, \dots, n\}$ alors $v(E)$ désigne le gain de la coalition des joueurs de E . Exemple : $v(1, 3, 4)$ est le gain obtenu lorsque les joueurs 1, 3 et 4 se regroupent.
- On note $S = v(1, 2, \dots, n)$ le gain obtenu par le groupe tout entier. C'est cette somme qu'il faut maintenant partager équitablement entre tous les joueurs.
- Par définition $v(\emptyset) = 0$ (où \emptyset est l'ensemble vide, avec $\text{Card } \emptyset = 0$).

Hypothèse. Nous supposons que la fonction v est **sur-additive**, c'est-à-dire :

$$v(E \cup F) \geq v(E) + v(F),$$

pour toutes parties E et F de $\{1, \dots, n\}$ qui sont disjointes (c'est-à-dire $E \cap F = \emptyset$). Cela signifie que les joueurs de E et F réunis ensemble gagnent plus (ou autant) que la somme des gains de la coalition de E avec les gains de la coalition de F . Les joueurs ont donc intérêt à se réunir.

Formule de Shapley.

Voici une formule qui permet de répartir les gains σ_i en tenant compte de la valeur de chaque joueur i (plus le joueur i est important, plus son gain σ_i sera grand). La formule nécessite de connaître le gain $v(E)$ pour n'importe quel sous-ensemble E de $\{1, \dots, n\}$.

$$\sigma_{i_0} = \frac{1}{n} \sum_{\substack{E \subset \{1, \dots, n\} \\ \text{t.q. } i_0 \notin E}} \binom{n-1}{\text{Card } E}^{-1} (v(E \cup \{i_0\}) - v(E))$$

La formule est délicate à comprendre et on verra des exemples juste après. Notons que :

- $v(E \cup \{i_0\}) - v(E)$ est le bénéfice de voir i_0 rejoindre la coalition E ,
- la formule est une moyenne (pondérée) de tous ces bénéfices et prenant tous les sous-ensembles E de $\{1, \dots, n\}$ ne contenant pas i_0 (il ne faut pas oublier $E = \emptyset$ parmi ceux-ci).
- Rappelons que $\text{Card } E$ est le **cardinal** de l'ensemble E , c'est-à-dire son nombre d'éléments.
- $\binom{p}{k}$ est le **coefficient binomial** appelé aussi **k parmi p** et se calcule par la formule :

$$\binom{p}{k} = \frac{p!}{k!(p-k)!}.$$

Ainsi :

$$\binom{p}{0} = 1 \quad \binom{p}{1} = p \quad \binom{p}{2} = \frac{p(p-1)}{2} \quad \dots \quad \binom{p}{p} = 1$$

Exemples à deux joueurs.

Pour $n = 2$ la somme à partager est $S = v(1, 2)$. Les gains de chacun sont :

$$\sigma_1 = \frac{1}{2}(v(1, 2) + v(1) - v(2)) \quad \sigma_2 = \frac{1}{2}(v(1, 2) + v(2) - v(1))$$

Preuve pour σ_1 : les ensembles de $\{1, 2\}$ qui ne contiennent pas 1 sont $E = \emptyset$ (de cardinal 0, avec $v(\emptyset) = 0$) et $E = \{2\}$ (de cardinal 1) donc la formule de Shapley donne :

$$\sigma_1 = \frac{1}{2} \left(\underbrace{\binom{1}{0}^{-1} (v(1) - 0)}_{\text{cas } E=\emptyset} + \underbrace{\binom{1}{1}^{-1} (v(1, 2) - v(2))}_{\text{cas } E=\{2\}} \right) = \frac{1}{2} (v(1, 2) + v(1) - v(2))$$

Voici un exemple concret : supposons $v(1) = 4$, $v(2) = 2$, et pour les deux joueurs réunis $v(1, 2) = 10$ (ainsi $v(1, 2) \geq v(1) + v(2)$ comme requis pour la formule). Comment les joueurs se partagent le gain $S = 10$ obtenu en commun ? La formule de Shapley donne :

$$\sigma_1 = \frac{1}{2}(10 + 4 - 2) = 6 \quad \text{et} \quad \sigma_2 = \frac{1}{2}(10 + 2 - 4) = 4$$

Ainsi même si le joueur 1 est individuellement deux fois plus fort que le joueur 2, il n'obtient pas deux fois plus que le joueur 2, en effet il a eu besoin du joueur 2 pour partager une plus forte somme qu'il n'aurait pas pu obtenir tout seul.

Terminons avec un autre exemple : un patron (joueur 1) et un employé (joueur 2). L'employé ne peut pas travailler si le patron n'est pas là. Ainsi la fonction valeur est la suivante :

$$v(1) = 1 \quad v(2) = 0 \quad v(1, 2) = 2$$

La formule de Shapley nous donne $\sigma_1 = \frac{3}{2}$ et $\sigma_2 = \frac{1}{2}$. Ainsi la patron récupère 75% des bénéfices et donne 25% à l'employé. Noter que l'on n'est pas loin du pourcentage considéré comme acceptable dans le jeu de l'ultimatum.

Exemples à trois joueurs.

La formule de partage pour trois joueurs est :

$$\begin{aligned} \sigma_1 &= \frac{1}{6} (2v(1, 2, 3) + v(1, 2) + v(1, 3) + 2v(1) - 2v(2, 3) - v(2) - v(3)) \\ \sigma_2 &= \frac{1}{6} (2v(1, 2, 3) + v(1, 2) + v(2, 3) + 2v(2) - 2v(1, 3) - v(1) - v(3)) \\ \sigma_3 &= \frac{1}{6} (2v(1, 2, 3) + v(1, 3) + v(2, 3) + 2v(3) - 2v(1, 2) - v(1) - v(2)) \end{aligned}$$

À vous de vérifier ces formules, par exemple pour calculer σ_1 on considère les ensembles E ne contenant pas 1 : ce sont $\emptyset, \{2\}, \{3\}, \{2, 3\}$.

Considérons un exemple à trois joueurs avec $v(1) = 1$, $v(2) = 2$ et $v(3) = 3$. Supposons que lorsque les joueurs forment un groupe alors les gains s'ajoutent (la fonction v est additive $v(E \cup F) = v(E) + v(F)$, par exemple $v(1, 3) = v(1) + v(3) = 4$ et $v(1, 2, 3) = v(1) + v(2) + v(3) = 6$). Il n'y a donc pas de bonus à se regrouper. Vérifier que, dans ce cas, chacun récupère un gain selon sa valeur individuelle : $\sigma_1 = 1$, $\sigma_2 = 2$, $\sigma_3 = 3$.

Considérons l'exemple avec les données suivantes :

$$v(1) = 1 \quad v(2) = 2 \quad v(3) = 0 \quad v(1, 2) = 4 \quad v(1, 3) = 2 \quad v(2, 3) = 4 \quad v(1, 2, 3) = 10$$

Alors

$$\sigma_1 = 3 \quad \sigma_2 = 4.5 \quad \sigma_3 = 2.5$$

Noter que dans cet exemple le joueur 3 n'obtient aucun gain individuellement mais qu'il apporte une contribution lorsqu'il est en groupe. Noter aussi aussi que le fait de se regrouper apporte un bonus notable.

Terminons avec un patron fainéant (qui ne produit pas) et ses deux employés (qui ne produisent qu'en sa présence) :

$$v(1) = 0 \quad v(2) = 0 \quad v(3) = 0 \quad v(1, 2) = 1 \quad v(1, 3) = 1 \quad v(2, 3) = 0 \quad v(1, 2, 3) = 2$$

Alors

$$\sigma_1 = 1 \quad \sigma_2 = \frac{1}{2} \quad \sigma_3 = \frac{1}{2}$$

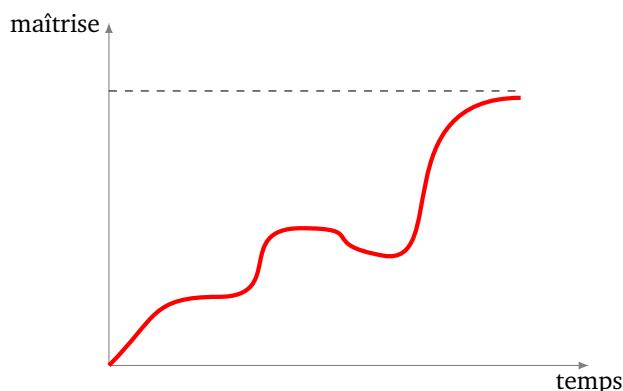
Ainsi le patron fainéant récupère la moitié des gains. On retrouve ce résultat plus généralement pour un patron fainéant et k employés : le patron récupère la moitié des gains et ses employés se partagent l'autre moitié.

2. Qu'est-ce qu'un « bon » jeu

Un bon jeu c'est de l'amusement et de l'engagement ! Voyons quelques paramètres pour créer un tel jeu.

2.1. Courbe d'apprentissage

Nous allons voir différentes façons de décrire l'apprentissage d'un joueur.



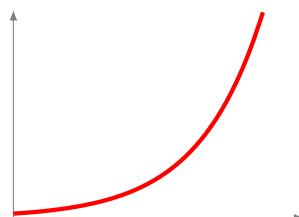
En abscisse nous notons « temps » parce que c'est le principal facteur d'apprentissage : la durée passée à apprendre. Ce n'est pas toujours vrai (par exemple si on est tout seul, on peut apprendre de travers). On pourrait parler d'« expérience » mais ce mot reste ambigu. En ordonnée nous avons la « maîtrise », cela pourrait aussi être la « connaissance » ou bien le « niveau » atteint. Cette valeur peut être limitée ou pas. Une analogie économique serait la courbe des « profits » (ordonnée) en fonction de l'« investissement » (abscisse).

Voici différents modèles de courbes d'apprentissage.



Fonction linéaire

$$f(x) = kx$$



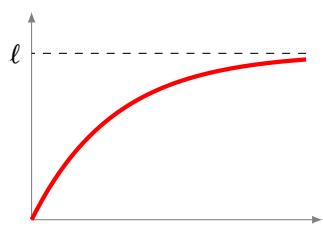
Fonction exponentielle

$$f(x) = e^x$$



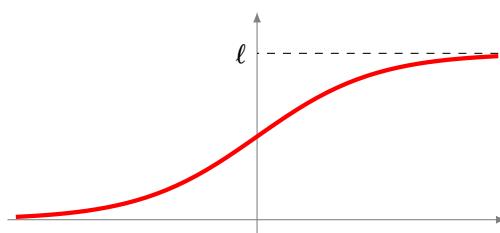
Fonction logarithme

$$f(x) = \ln(x + 1)$$



Amorti exponentiel

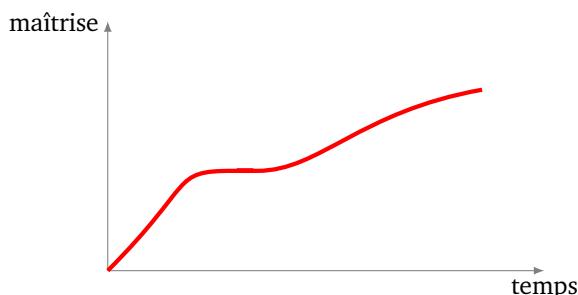
$$f(x) = l(1 - e^{-x})$$



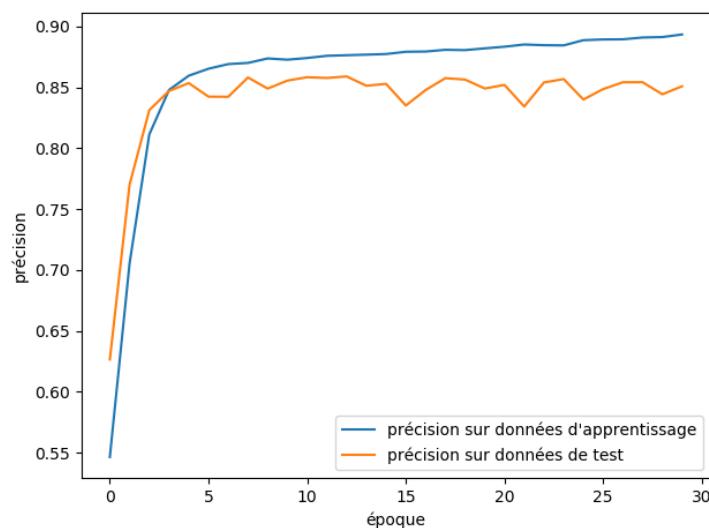
Fonction sigmoïde

$$f(x) = \frac{l}{1+e^{-x}}$$

On peut distinguer les courbes qui sont bornées (sur la première ligne des figures précédentes) ou non (sur la seconde ligne). On peut aussi distinguer les apprentissages à feedback positif ou négatif. Une évolution à feedback positif : plus j'avance, plus mon niveau augmente vite. Ce sont des courbes qui croissent plus vite qu'une fonction linéaire (c'est par exemple le cas avec une courbe exponentielle). Par exemple, au monopoly si j'ai peu de propriétés je gagne peu d'argent, mais si j'ai beaucoup de propriétés je gagne beaucoup beaucoup d'argent et je peux acheter davantage de propriétés. Une évolution peut aussi être à feedback négatif : plus j'avance plus les niveaux sont difficiles à atteindre. Par exemple : je termine un niveau, mais le niveau suivant sera plus dur, ou alors je doit atteindre le même objectif mais avec moins de ressources. C'est par exemple le cas pour la progression d'une armée (réelle ou virtuelle), plus elle s'enfonce en territoire ennemi, plus sa progression est difficile (ravitaillement, communication, connaissance du terrain...). Dans la vie réelle, l'apprentissage (par exemple de la conduite d'une voiture) connaît des périodes de stagnation (voire de régression) :



La notion de courbe d'apprentissage ne concerne pas que les humains. Les ordinateurs aussi apprennent : par exemple à distinguer un chat d'un chien sur une photo, à jouer aux échecs et à piloter des voitures. Pour cela il s'agit d'entraîner un « réseau de neurones » en lui montrant par exemple 1000 images de chiens et 1000 images de chats (et en lui disant qui est qui). Ce processus est itéré plusieurs fois avec les mêmes images. Ensuite un réseau de neurones bien entraîné saura reconnaître de nouvelles images. Voici typiquement la courbe d'apprentissage : elle mesure le taux de succès (par exemple reconnaissance correcte de chat vs chien sur des *nouvelles* images) en fonction de la longueur d'apprentissage (on parle d'« époques ») (par exemple le nombre d'images soumises, sachant qu'une fois les 2000 images épuisées on recommence avec la première image).



La précision sur les données d'apprentissage augmente et finit par plafonner. Sur les données de test la précision augmente aussi puis l'apprentissage ne sert plus à rien et la précision peut même diminuer. Que se passe-t-il ? C'est un phénomène de sur-apprentissage. Le réseau de neurones a fini par apprendre par cœur les données d'entraînement mais n'est plus efficace sur les données nouvelles (comme un étudiant qui apprend par cœur les solutions de la feuille d'exercices sans comprendre : il n'arrivera pas à faire les exercices inédits le jour de l'examen !).

2.2. Jeu gagnable

Un bon jeu se doit d'être « gagnable » et « équitable ».

Gagnable.

- Le jeu doit être perçu comme gagnable, ceci quelle que soit la progression du joueur dans le jeu.
- On peut découper le jeu en niveaux afin de proposer des gains partiels et des récompenses intermédiaires.
- Le joueur doit sentir qu'il progresse vers l'objectif.
- Un joueur expérimenté doit avoir plus de chance de gagner.
- Le jeu doit pouvoir être gagné même en cas d'une erreur du joueur au début de la partie.

Pas trop facile.

- Un jeu ne doit pas être trop facile, sinon il perd de son intérêt.
- Cependant les débuts doivent être faciles pour éviter de décourager le joueur.
- Les jeux toujours faciles et à priori sans intérêt peuvent avoir du succès, ils permettent par exemple de se vider la tête ou de s'occuper sans réfléchir.

Illusion de gagnabilité.

- Le jeu doit paraître gagnable mais ne l'est pas forcément. Autrement dit, même s'il est très rarement possible de gagner, le joueur doit toujours penser que c'est possible.
- Un bon exemple est ce jeu de fête foraine dans lequel on glisse des pièces qui entraînent d'autres pièces qui sont censées finir par tomber par gros paquets, mais où en fait on ne récupère au mieux que quelques pièces.
- Un autre exemple est le loto. Si le loto était présenté comme un tirage d'un numéro entre un et dix millions, personne ne jouerait. Mais avec le système à plusieurs numéros, le joueur va régulièrement obtenir un ou plusieurs bons numéros, voire des numéros proches des bons, ce qui lui fera penser qu'il n'était pas loin de gagner (notion qui n'a pas de sens ici !). De plus la loterie médiatise fortement les succès des gagnants pour obtenir un effet « pourquoi pas moi ? »

2.3. Jeu équitable

Équité.

Que le jeu soit un jeu de :

- *compétence* (échec, course de voitures, jeu de tir...),
- *patience* (dressage d'animaux, gestion d'une ferme, peinture au numéro...),
- *chance* (loterie, casino...),

alors ce jeu doit être équitable.

Voici la définition de « équité » dans le Larousse :

Équité : Qualité consistant à attribuer à chacun ce qui lui est dû par référence aux principes de la justice naturelle.

Il ne faut pas confondre *équité* et *égalité* : il semble normal lors d'une récompense d'attribuer une plus grande part au plus gros contributeur de la réussite.

Ressources.

- Les ressources (objets, pouvoirs, terrain, argent, temps...) doivent être partagées de façon équitable (ou même égale) entre les joueurs.
- Une façon de réaliser ceci est par symétrie du jeu. Par exemple aux échecs les deux joueurs ont les mêmes pièces avec les mêmes mouvements. La seule petite asymétrie (très importante pour les grands joueurs) est que le joueur avec les pièces blanches commence. Ce défaut de symétrie est réparé sur plusieurs parties en alternant celui qui débute la partie.
- Pour les jeux asymétriques on peut rétablir l'équilibre par des relations de non-transitivité. Rappelons qu'une relation est *transitive* si $a \mathcal{R} b$ et $b \mathcal{R} c$ implique $a \mathcal{R} c$ (exemple : si $a \leq b$ et $b \leq c$ alors $a \leq c$). Exemple de relation non-transitive : les joueurs peuvent choisir entre trois personnages *A*, *B* ou *C* qui ont chacun des pouvoirs différents. Pour que le jeu soit intéressant il ne faut pas qu'un personnage soit plus fort que tous les autres. Il faut par exemple que $A \geq B$, $B \geq C$ mais $C \geq A$. C'est le cas dans le jeu « pierre/feuille/ciseau », « poules/renards/vipères » ou pour les cartes *Pokémon*.
- Le ratio coût/avantage des objets doit être bien dosé. Exemple : obtenir un super-pouvoir doit demander de battre un monstre vraiment méchant.
- Les ressources doivent être méritées si elles ne sont pas partagées équitablement. Exemple : l'achat intégré d'un joli costume pour un avatar ne pose pas de problème, par contre l'achat d'une super-arme qui permet de tout gagner est mal venu.

Niveau.

- Un joueur plus expérimenté doit avoir plus de chance de gagner.
- Pour conserver le plaisir de jouer on peut regrouper les joueurs par poules de niveau équivalent.
- Si on décide de jouer une partie avec des joueurs de différents niveaux, alors on peut attribuer un *handicap* aux meilleurs joueurs. Par exemple les meilleurs chevaux d'une course hippique sont chargés d'un poids supplémentaire. Au contraire on peut aussi favoriser les joueurs les moins bons. Par exemple au *go*, le joueur le moins bon a des pierres supplémentaires au départ ; à *Mario Kart* le dernier de la course trouve sur sa route des boîtes qui contiennent des avantages conséquents qui lui permettent de revenir dans la course.

Contrôle.

- Les décisions du joueur doivent être déterminantes pour l'issue du jeu.
- Par exemple si vous devez choisir entre le chemin de gauche contre un monstre ou celui de droite vers un lac paisible cela doit avoir une conséquence pour la suite du jeu (par exemple gain d'expérience, perte de vie, chemins ultérieurs possibles...). Cependant aucun choix (surtout en début de partie) ne devrait empêcher de gagner la partie.

Chance.

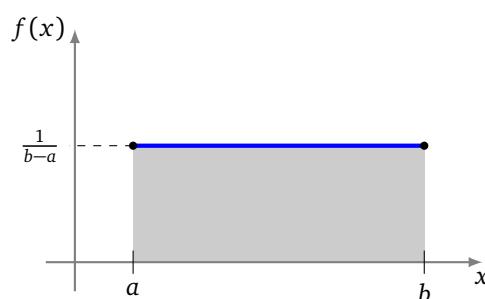
- Le hasard c'est lorsque le joueur n'a pas de contrôle sur l'issue de l'aléa.
 - Un jeu de hasard ou avec une part de chance peut quand même être un jeu équitable, mais sous certaines conditions.
 - Le gain, ainsi que la probabilité de le gagner, doivent être connus du joueur. Par exemple, pour atteindre la rue de la Paix je dois faire 7 avec deux dés : je sais que j'aurai une chance sur six. De même au loto ou au casino le jeu est considéré comme équitable car le gain et la probabilité de gagner sont publics (en fait le loto et le casino redistribuent entre 50% et 95% des mises).
 - Un exemple de ce qui n'est pas équitable peut être des *loot boxes* (ce sont des boîtes surprises que l'on peut acheter au cours du jeu). Le joueur ne sait pas toujours s'il va obtenir un nouveau costume pour son avatar (sans intérêt) ou bien une nouvelle arme (qui le rend plus puissant).
 - Un jeu peut avoir une part de chance sous la forme :
 - un peu de hasard de façon assez fréquente,
 - un gros aléa mais qui se produit rarement.
- On peut aussi laisser le choix au joueur de faire intervenir le hasard : « payer une amende de 100 euros ou bien tirer une carte chance ».
- Lorsqu'il est crucial pour le jeu, le hasard devrait être *certifié*. C'est particulièrement important pour les jeux d'argent en ligne (poker, black jack, roulette...) : le joueur doit être sûr que le tirage est vraiment aléatoire. On peut utiliser un générateur éprouvé de nombres pseudo-aléatoires pour lequel les différents tirages n'ont aucun lien entre eux.

Lois.

Terminons par deux types de lois qui permettent de générer du hasard :

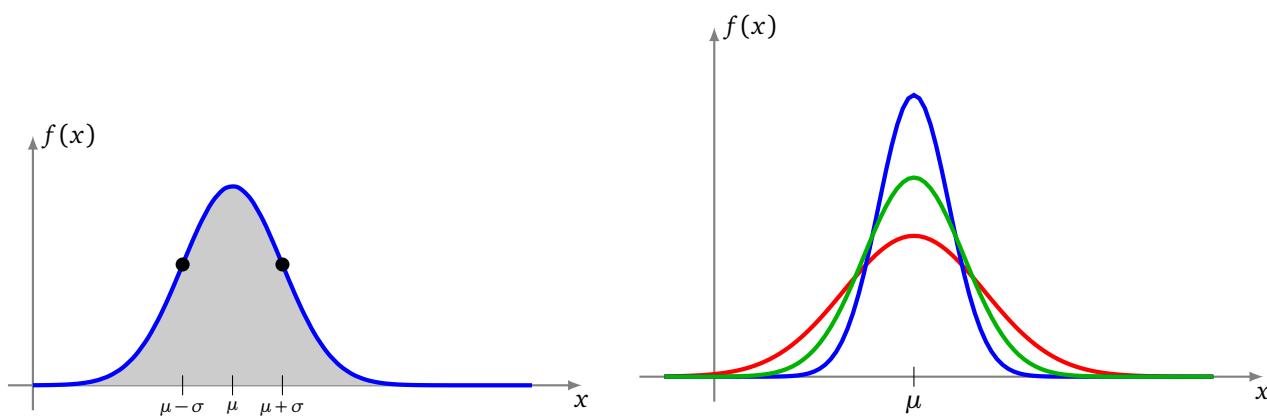
- la **loi uniforme** : par exemple chaque nombre entre a et b a autant de chance que les autres d'être choisi (exemple : tirage du loto) ;
- la **loi normale**, dite aussi **loi de Gauss** ou **courbe en cloche**, pour laquelle les valeurs autour d'une valeur μ ont plus de chance d'être choisies que les valeurs éloignées. C'est par exemple la situation de la somme de deux dés (ou la plus forte probabilité est d'obtenir 7 alors qu'obtenir 12 est moins probable), ou la taille d'un homme choisi au hasard dans la population...

La densité de probabilité pour la loi uniforme sur $[a, b]$ est la fonction définie par $f(x) = \frac{1}{b-a}$.



La densité de probabilité de la loi normale d'espérance μ et d'écart-type σ est :

$$f(x) = \frac{1}{\sqrt{2\pi\sigma^2}} \exp\left(-\frac{1}{2} \frac{(x-\mu)^2}{\sigma^2}\right)$$



Sur la figure de gauche, on représente le graphe de f pour une valeur donnée de (μ, σ) ; la densité de probabilité diminue lorsqu'on l'on s'éloigne de μ ; l'aire sous la courbe vaut 1. Sur la droite sont tracés les graphes de densité de probabilité de la loi normale pour une même valeur de μ mais avec différentes valeurs σ qui mesurent l'étalement de la courbe.

2.4. Candy Crush

Détaillons quelques aspects d'un jeu à succès : *Candy Crush*. Ceux qui ne connaissent pas peuvent le tester avant de lire la suite, mais attention de ne pas devenir accro ! On résume ici l'étude *Deconstructing Candy Crush : what instructional design can learn from game design* par E. Varonis et M. Varonis (*International Journal of Information and Learning Technology*, 2015).

A. Structure du jeu.

1. Alignement de bonbons.

- Le joueur doit intervertir deux pièces du jeu (des bonbons colorés) afin d'en aligner trois du même type qui vont alors être retirés de la grille. Les bonbons situés au-dessus descendent d'un cran, créant éventuellement d'autres alignements... Un alignement de quatre bonbons ou plus crée un bonbon spécial.
- Ce jeu est joué chaque jour par des millions de personnes dans le monde. Même s'il est gratuit, il produit de très gros bénéfices via des achats intégrés.

2. Jeu accessible et addictif.

- *Candy Crush* est jouable n'importe où, n'importe quand : il est multi-plateformes et réussir un niveau prend quelques minutes.
- L'univers bienveillant ramène à l'enfance. Les couleurs sont vives, les enchainements d'explosions en cascades sont inattendus.

3. Niveaux.

- Le jeu est découpé en niveaux (plus de 10 000) regroupés en épisodes. La carte de progression n'affiche que quelques prochains niveaux pour ne pas étourdir le joueur par l'ampleur du chemin.
- Chaque niveau a un objectif clair parmi plusieurs possibilités (atteindre un score, faire disparaître de la gélatine, faire tomber certains bonbons...). Le nombre de coups est limité (de façon bien dosée).
- Les niveaux sont de difficulté croissante. Il n'y a pas de limite pour recommencer un niveau (à condition de patienter si on n'a plus de vies). Les nouvelles difficultés et fonctionnalités apparaissent progressivement.
- Il n'y a pas de règles écrites, ni manuel, ni tutoriel. L'apprentissage se fait par la pratique. Cependant une aide existe pour trouver les coups possibles afin que le joueur ne soit pas bloqué tant qu'il lui reste des coups possibles.

4. Feedback immédiat

- Retour très positif dès la victoire (message, musique, bonus...).

- En cas d'échec, le jeu encourage à utiliser des coups supplémentaires (en utilisant ses bonus ou par achat).

5. Chance.

- La configuration initiale est aléatoire. Le joueur ne peut pas prévoir les conséquences complètes de son mouvement qui peut engendrer (ou pas) une cascade d'alignements.
- Le facteur aléatoire incite à recommencer en cas d'échec en espérant un meilleur déroulement de la partie.

6. Temps.

- Certains niveaux sont à temps limité.
- Il y a un délai d'attente pour récupérer des vies. Cela peut engendrer de la frustration et incite à acheter des vies supplémentaires, il y a aussi des activités annexes pour patienter.

7. Social.

- Le partage des succès est une publicité efficace pour le jeu et apporte une fierté personnelle. Cela permet aussi un challenge bon enfant entre amis.
- L'entraide entre amis (échange de vies, de bonus...) fait à la fois plaisir à celui qui reçoit mais aussi à celui qui donne.

Conclusion. Un bon jeu mobile doit avoir au moins deux des caractéristiques parmi : (a) des règles simples, (b) des interactions sociales, (c) pas d'ennemis à combattre. *Candy Crush* remplit ces trois conditions !

B. Aspect cognitif et émotionnel.

1. Jeu complexe.

- *Candy Crush* nécessite des capacités cognitives élevées (reconnaissance de formes, visualisation spatiale, coordination œil/main...).
- Le joueur doit deviner les règles et les lois du jeu dès l'apparition de nouveaux éléments. La difficulté est accentuée par des violations de certaines lois de la physique (par exemple, un bonbon en bas d'une colonne se retrouve ensuite tout en haut).
- *Candy Crush* est un problème de la catégorie NP-difficile (donc il n'y a pas d'algorithme connu qui trouve une solution en temps polynomial).
- Les niveaux sont globalement de difficulté croissante avec quelques niveaux éparses plus faciles qui permettent une respiration et maintiennent l'amusement.
- Le joueur doit garder en tête de nombreux paramètres : l'objectif bien sûr, mais aussi les ressources disponibles (nombre de coups restants, temps, bonus...)

2. Le joueur doit innover.

- Le jeu évolue au fil des niveaux et le joueur doit innover par itérations progressives. Les nouveautés sont fréquentes à la fois pour les objectifs du niveau et pour sa résolution. Cependant il n'y a pas de grand saut de difficulté pour éviter les blocages et la frustration.
- Le jeu nécessite force et finesse : la stratégie à adopter est différente d'un niveau à l'autre. Il faut choisir le bon dosage stratégie/risque : prévoir longuement ses coups ou bien jouer vite quitte à perdre et à recommencer le niveau.

3. Univers positif.

- L'univers du jeu est coloré et enfantin (bonbons, couleurs vives, explosions et cascades surprises, messages encourageants...).
- Le but est la libération de dopamine (un neurotransmetteur du cerveau) qui donne un fort sentiment de satisfaction.
- Le joueur a le sentiment d'être une personne performante et privilégiée car il reçoit de nombreuses récompenses.

4. Récompenses et réussites intermittentes.

- Les récompenses et les victoires ne sont pas systématiques. Certains niveaux sont délibérément plus difficiles et de toute façon le jeu a une part de chance (position initiale et conséquence des coups).
- Cela explique le comportement addictif de certains joueurs (qui finissent par dépenser des sommes énormes), comme certains joueurs de casino.
- Ceux qui échouent veulent retenter, ceux qui gagnent veulent continuer.

5. Frustration.

- En cas d'échec, le jeu est conçu pour donner le sentiment d'avoir « presque réussi » et propose d'acheter des coups supplémentaires afin de poursuivre le niveau en cours.
- L'investissement en temps et en argent pousse le joueur à continuer car il se dit « puisque j'ai passé beaucoup de temps/dépensé beaucoup d'argent, je ne vais pas m'arrêter maintenant ».

Conclusion. Candy Crush est un jeu conçu pour être détendant, positif et ne demande pas trop de réflexion. Il place le joueur dans un *flow* par une succession de niveaux à la difficulté bien ajustée qui valorisent les compétences du joueur avec des récompenses immédiates. Une part importante de chance et des niveaux volontairement plus difficiles entraînent frustration et addiction. Un des créateurs du jeu conclut : « L'objectif de la création d'un niveau est de trouver le bon mélange entre plaisir et souffrance. »

Notes et bibliographie

Mathématiques

Vous trouverez un cours complet d'algèbre et d'analyse avec exercices et vidéos ici :

exo7.emath.fr

Toutes les ressources ci-dessous sont en anglais.

Mathématiques et jeux vidéo

Voici quelques chaînes proposant des vidéos intéressantes :

- www.youtube.com/@g5min
- www.youtube.com/@codingmath
- www.youtube.com/@JorgeVinoRodriguez
- www.youtube.com/@Acephikmo

Graphisme

- www.scratchapixel.com : un site très clair qui explique les principes fondamentaux pour générer l'image d'une scène 3D.
- *Mathematics for 3D Game Programming and Computer Graphics* de Lengyel (Course Technology) : tout est dans le titre !
- *Computational Geometry* de Berg, Cheong, van Kreveld et Overmars (Springer) et *Discrete and Computational Geometry* de Devadoss et O'Rourke (Princeton University Press) : deux livres de géométrie algorithmique à propos des graphes, des polygones, des triangulations.

Logiciels

- *Pygame* est idéal pour développer des petits jeux 2D avec *Python* : www.pygame.org
- *Blender* permet de créer des images 3D ainsi que des petites animations : www.blender.org

Remerciements

Je remercie Stéphanie Bodin et Michel Bodin pour leurs relectures. Merci à Kroum Tzanez pour certaines figures.

Vous pouvez récupérer l'intégralité des codes *Python* ainsi que tous les fichiers sources sur la page *GitHub* d'Exo7 : « [GitHub : Mathgame](https://github.com/Exo7-Team/Mathgame) ».



Ce livre est diffusé sous la licence *Creative Commons – BY-NC-SA – 4.0 FR*.
Sur le site Exo7 vous pouvez télécharger gratuitement le livre en couleurs.

Index

- accélération, 215
- algorithme
 - de Bresenham, 117
 - de Catmull-Clark, 209
 - de coloriage, 126
 - de l'enroulement, 189
 - de triangulation, 192, 198
 - du peintre, 158
 - du *ray-tracing*, 173
 - du z-buffer, 157
 - élagage alpha-beta, 315
 - méthode d'Euler, 243
 - méthode de Runge-Kutta, 247
 - minimax, 305
 - parcours en largeur, 302
 - parcours en profondeur, 302
- angle
 - d'Euler, 58
 - degrés, 2
 - radians, 2
- animation, 128
- anticrénelage/*antialiasing*, 121
- approximation
 - de Padé, 241
 - de Tchebychev, 238
 - développement limité, 227
- arctan2, 6
- arctangente, 6
- base, 28
 - canonique, 28
 - orthonormale, 47
- boîte englobante, 178
- cellule de Voronoï, 199
- cercle trigonométrique, 2
- coordonnées
- barycentriques, 141
- cylindriques, 14, 78
- homogènes, 48
- polaires, 8
- sphériques, 15, 79
- couleur, 100
- courbe
 - cubique de Bézier, 231
 - d'apprentissage, 325
 - de Bézier d'ordre n , 236
 - de Hilbert, 218, 275
 - de Jordan, 88
 - quadratique de Bézier, 206
- déterminant, 26, 39, 44, 177
- dilemme du prisonnier, 293
- distance, 27
- enveloppe convexe, 189
- équation différentielle, 243
- équilibre de Nash, 297
- formule
 - d'Euler, 185
 - de Rodrigues, 57
 - du partage de Shapley, 322
- fractale, 255
- IFS, 264
- L-système, 269
- frustrum*, 162
- graphé, 181
 - arbre, 182, 301
 - planaire, 184
- homographie, 150
- homothétie, 14, 38, 41
- illumination, 100

de Blinn-Phong, 112
 de Phong, 105
 interpolation
 bilinéaire, 147
 circulaire, 205
 de Lagrange, 236
 linéaire, 205
 labyrinthe, 218
 loi
 de Gauss, 329
 de Snell-Descartes, 171, 283
 uniforme, 329
 lumière, 101
 réflexion, 170, 282
 réfraction, 171, 283
 luminosité, 100
 ambiante, 106
 diffuse, 107
 spéculaire, 109
 maillage, 203
 matrice, 30
 carrée, 35
 de gains, 293
 de passage, 45
 diagonale, 36
 identité, 35
 inverse, 40
 orthogonale, 47
 produit, 33
 rotation, 32
 transposée, 37
 triangulaire, 36
 nombre complexe, 12
 norme, 18, 27
 onde, 280
 perspective
 atmosphérique, 128
 axonométrique, 75
 linéaire, 68
 pixel, 113, 159
 produit
 mixte, 26
 scalaire, 19, 27
 vectoriel, 24
 projection, 39, 43
 cylindrique, 78
 orthogonale, 70
 parallèle, 74
 sphérique, 79
 quaternion, 63
ray tracing, 83, 157
 rayon d'ombre, 166
 réflexion, 13, 38, 43
 règle de Cramer, 177
 rotation, 14, 38, 42, 54
 blocage de cadran, 61
 lacet, 59, 273
 roulis, 59, 273
 tangage, 59, 273
 texture, 132
 bump map, 136
 displacement map, 138
 normal map, 138
 théorème
 de Pohlke, 77
 transformation
 affine, 44, 50, 265
 vectorielle, 44
 translation, 13, 38, 41
 transparence, 170
 triangulation, 187
 basculement, 198
 de Delaunay, 195
 vecteur, 17, 23
 coordonnées, 28, 45
 espace vectoriel, 26
 normal, 21
 nul, 27
 tangent, 233
 unitaire, 18
 vitesse, 214