

# Introduction à l'algorithmique : Ouverture

Serge Haddad

LSV, ENS Paris-Saclay & CNRS & Inria

L3

# Tests géométriques

Soient  $p_1, p_2, p_3 \in \mathbb{R}^2$  des points non alignés et  $p$  un point quelconque.

- $p$  appartient au triangle formé par  $p_1, p_2$  et  $p_3$   
ssi l'unique solution  $(\lambda_1, \lambda_2, \lambda_3)$  de :

$$\sum_{i \leq 3} \lambda_i p_i = p \quad \wedge \quad \sum_{i \leq 3} \lambda_i = 1$$

appartient à  $(\mathbb{R}^+)^3$ .

- $p$  appartient au cône d'origine  $p_1$ , et de directions  $\overrightarrow{p_1 p_2}$  et  $\overrightarrow{p_1 p_3}$   
ssi l'unique solution  $(\lambda_2, \lambda_3)$  de :

$$\lambda_2 \overrightarrow{p_1 p_2} + \lambda_3 \overrightarrow{p_1 p_3} = \overrightarrow{p_1 p}$$

appartient à  $(\mathbb{R}^+)^2$ .

Ces tests s'opèrent en temps constant.

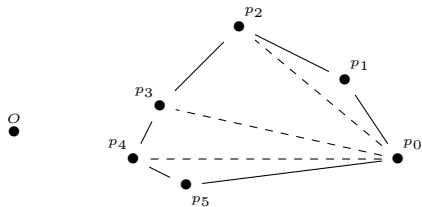
# Polygone convexe

Un polygone  $\mathcal{P}$  est donné par une liste  $p_0, \dots, p_{n-1}$  de ses sommets tels que la liste  $(\overline{p_i p_{i+1 \% n}})_{0 \leq i < n}$  soit celle de ses segments parcourue en sens direct.

Il est convexe ssi tous les angles vus de l'intérieur sont saillants.

Test (naïf) d'appartenance en  $O(n)$

$$O \in \mathcal{P} \text{ ssi } \exists 0 < i < n - 1 \ O \in \text{Triangle}(p_0, p_i, p_{i+1 \% n})$$



# Ajout d'un sommet

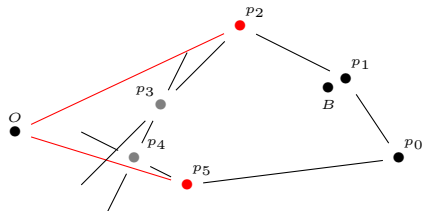
Soit  $O \notin \mathcal{P}$ , polygone convexe.

- ▶ Calcul de  $B = \frac{1}{3}(p_0 + p_1 + p_2)$  ;
- ▶ Pour tout  $i < n$ , extraction de  $p_i$  ssi :

$$O \in \text{Cone}(p_i, \overrightarrow{p_{i-1\%n}p_i}, \overrightarrow{p_{i+1\%n}p_i})$$

- ▶ Parmi les  $n'$  points restants, insertion de  $O$  entre  $p_{\alpha(i)}$  et  $p_{\alpha(i+1\%n')}$  tels que :

$$O \in \text{Cone}(B, \overrightarrow{Bp_{\alpha(i)}}, \overrightarrow{Bp_{\alpha(i+1\%n')}})$$



# Enveloppe convexe

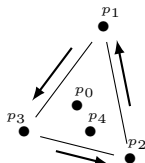
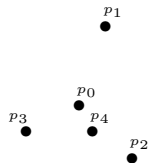
Soit  $S$  un ensemble de  $n \geq 3$  points  $p_0, \dots, p_{n-1}$  dans le plan.

$EC(S)$ , l'enveloppe convexe de  $S$ , est définie par :

$$EC(S) = \{p \mid \exists \{\lambda_i\}_{i < n} \in (\mathbb{R}^+)^n \wedge \sum_{i < n} \lambda_i = 1 \wedge p = \sum_{i < n} \lambda_i p_i\}$$

$EC(S)$  est un polygone (convexe)

dont l'ensemble des sommets  $S'$  est un sous-ensemble de  $S$ .



**Objectif.** Construire une représentation de  $EC(S)$ .

- ▶ Algorithme de Graham
- ▶ Algorithme dichotomique
- ▶ Algorithme de Jarvis
- ▶ Algorithme de Chan

# Intérieur de l'enveloppe convexe

$EC^\circ(S)$ , l'intérieur de  $EC(S)$ , est défini par :

$$EC^\circ(S) = \{p \mid \exists r > 0 \text{ Disc}(p, r) \subseteq EC(S)\}$$

$EC^\circ(S)$  est non vide si et seulement si il existe  $p_i, p_j, p_k$  non alignés.

Comment trouver un point de  $EC^\circ(S)$  ?

```
i ← 2;  
While i < n and  $p_1 = p_i$  do i ← i + 1;  
If  $i \geq n$  then return(false);  
j ← i + 1;  
While  $j \leq n$  and  $p_j \in d(p_1, p_i)$  do j ← j + 1;  
If  $j > n$  then return(false);  
return( $\frac{1}{3}(p_1 + p_i + p_j)$ );
```

# Circuit polaire

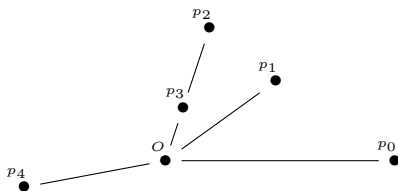
Soit deux vecteurs non nuls en coordonnées polaire  $\mathbf{v} = (\rho, \theta)$  et  $\mathbf{v}' = (\rho', \theta')$ .

$$\mathbf{v} \prec \mathbf{v}' \text{ ssi } \theta < \theta' \text{ ou } (\theta = \theta' \text{ et } \rho > \rho')$$

Soit  $O$  une *origine* et  $S = \{p_0, \dots, p_{n-1}\}$  un ensemble de points différents de  $O$ .

Soit  $\alpha$  une permutation.  $(p_{\alpha(0)}, \dots, p_{\alpha(n-1)})$  est un circuit polaire relatif à  $O$

si pour tout  $i < n$ ,  $\overrightarrow{Op_{\alpha(i)}} \prec \overrightarrow{Op_{\alpha(i+1)}}$  avec  $\overrightarrow{Op_{\alpha(0)}}$  comme axe  $\overrightarrow{Ox}$ .



Le circuit polaire s'obtient par un tri en  $O(n \log(n))$ .

# Eviter les coordonnées polaires

Soit  $O$  le point origine et  $\overrightarrow{Op_0}$  l'orientation de l'axe des abscisses.

$\text{Prec}(p, p')$

*(vecteurs de même direction)*

**If**  $\exists \lambda \overrightarrow{Op} = \lambda \overrightarrow{Op'} \wedge \lambda > 1$  **then return true**

**If**  $\exists \lambda \overrightarrow{Op} = \lambda \overrightarrow{Op'} \wedge 0 < \lambda \leq 1$  **then return false**

*(un vecteur de direction  $\overrightarrow{Op_0}$ )*

**If**  $\exists \lambda \overrightarrow{Op} = \lambda \overrightarrow{Op_0} \wedge \lambda > 0$  **then return true**

**If**  $\exists \lambda \overrightarrow{Op'} = \lambda \overrightarrow{Op_0} \wedge \lambda > 0$  **then return false**

*( $\overrightarrow{Op}$  appartient au demi-plan supérieur)*

**If**  $\det(\overrightarrow{Op_0}, \overrightarrow{Op}) > 0$  **then return**  $p' \notin \text{Cone}(\overrightarrow{Op_0}, \overrightarrow{Op})$

*( $\overrightarrow{Op}$  de direction opposée à  $\overrightarrow{Op_0}$ )*

**If**  $\det(\overrightarrow{Op_0}, \overrightarrow{Op}) = 0$  **then return**  $\det(\overrightarrow{Op_0}, \overrightarrow{Op'}) < 0$

*( $\overrightarrow{Op}$  appartient au demi-plan inférieur)*

**If**  $\det(\overrightarrow{Op_0}, \overrightarrow{Op}) < 0$  **then return**  $p' \in \text{Cone}(\overrightarrow{Op_0}, \overrightarrow{Op})$



# Du circuit polaire à l'enveloppe convexe

Soit  $\mathcal{C}$  un circuit polaire de  $n$  points (liste circulaire) avec  $O \in EC^{\circ}(\mathcal{C})$ .

Choisir  $p_0 \in \mathcal{C}$  extrémal pour un ordre lexicographique des coordonnées.

$p \leftarrow p_0$ ;

**While**  $\text{succ}(\mathcal{C}, p) \neq p_0$  **do**

**If**  $\text{succ}(\mathcal{C}, p) \in \text{Triangle}(O, p, \text{succ}(\mathcal{C}, \text{succ}(\mathcal{C}, p)))$  **then**

*Extraire*( $\mathcal{C}, \text{succ}(p)$ ); **If**  $p \neq p_0$  **then**  $p \leftarrow \text{pred}(\mathcal{C}, p)$ ;

**Else**

$p \leftarrow \text{succ}(\mathcal{C}, p)$ ;

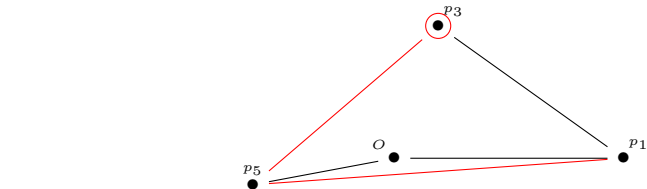
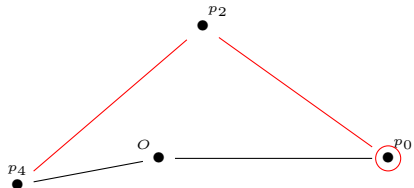
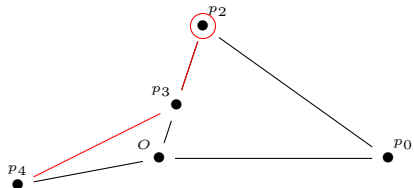
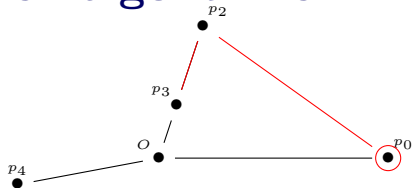
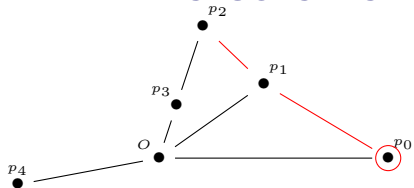
## Complexité.

Soit  $\ell = |\mathcal{C}| + \min(i > 0 \mid \text{succ}^{(i)}(p) = p_0)$ .

$\ell$  est initialement égal à  $2n$  et décroît à chaque étape.

D'où un algorithme en  $O(n)$ .

# Déroulement de l'algorithme



# Correction de l'algorithme

## Invariants.

- ▶ L'enveloppe convexe de  $EC(\mathcal{C})$  reste inchangée.
- ▶ Les angles formés par les segments de  $p_0$  à  $\text{succ}(\mathcal{C}, p)$  sont saillants (vus de  $O$ ).

## Extraction d'un sommet.

- ▶ Un sommet retiré appartient à l'intérieur de l'enveloppe convexe ou à l'intérieur d'une arête.
- ▶ Puisque  $p$  devient  $\text{pred}(\mathcal{C}, p)$ , la deuxième propriété reste vérifiée.

## Passage au successeur.

- ▶  $\mathcal{C}$  est inchangé.
- ▶ La deuxième propriété reste vérifiée par construction.

## Observation.

A la fin de l'algorithme l'angle en  $p_0$  est nécessairement saillant (choix de  $p_0$ ).

Un polygone (simple) dont tous les angles sont saillants est convexe.

# Diviser pour régner : le retour

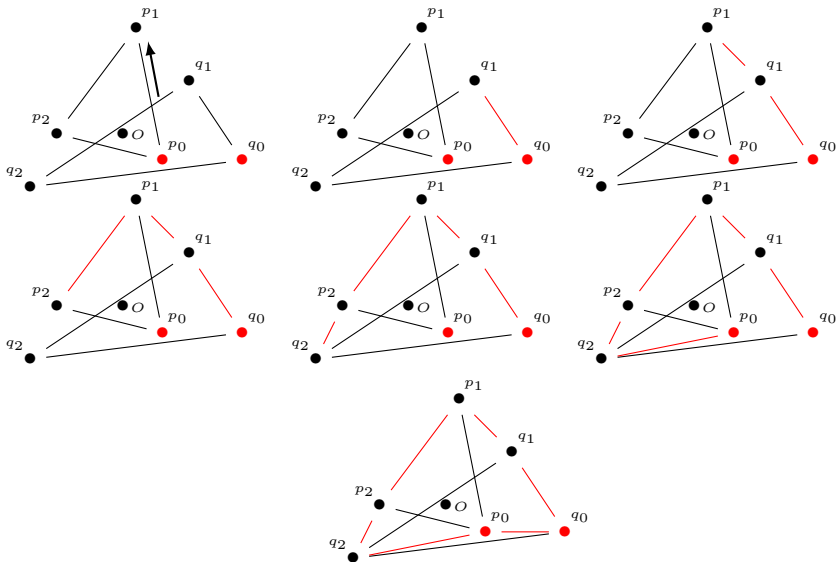
## Principe.

- ▶ Si  $n \leq 3$  alors éliminer les points redondants en  $O(1)$ .
- ▶ Répartir les points en 2 sous-ensembles de tailles approximativement égales.
- ▶ Calculer de manière récursive les polygones associés aux sous-ensembles :  $\mathcal{P} = (p_0, \dots, p_{r-1})$  et  $\mathcal{Q} = (q_0, \dots, q_{s-1})$
- ▶ Fusionner les polygones en  $O(r + s)$ .

## Fusion.

- ▶ Déterminer un point intérieur  $O$  à  $\mathcal{P}$  en  $O(1)$ .
- ▶ Si  $O \notin \mathcal{Q}$  alors ajouter  $O$  à  $\mathcal{Q}$  en  $O(s)$ .
- ▶ Déterminer le vecteur minimal  $\overrightarrow{Oq_i}$  relatif à  $\overrightarrow{Op_0}$  en  $O(s)$ .
- ▶ Fabriquer un circuit polaire à partir des listes  $p_0, \dots, p_{r-1}$  et  $q_i, \dots, q_{i-1\%s}$  (privé des points éliminés lors de l'ajout) en  $O(r + s)$ .
- ▶ Transformer le circuit polaire en polygone par la procédure de Graham.

# Des polygones au circuit polaire



# Algorithme de Jarvis

Soit  $r \in \mathcal{C}$  minimal pour l'ordre lexicographique des coordonnées  $(y, -x)$ .

Insérer( $r$ );

**If**  $r = p_0$  **then**  $p \leftarrow p_1$  **else**  $p \leftarrow p_0$ ;

**For**  $i$  **from** 0 **to**  $n - 1$  **do**

**If**  $p_i \neq r$  **and**  $\overrightarrow{rp_i} \prec \overrightarrow{rp}$  w.r.t.  $\overrightarrow{rx}$  **then**  $p \leftarrow p_i$ ;

Insérer( $p$ );

$q \leftarrow r$ ;

**Repeat**

**If**  $p = p_0$  **then**  $o \leftarrow p_1$  **else**  $o \leftarrow p_0$ ;

**For**  $i$  **from** 0 **to**  $n - 1$  **do**

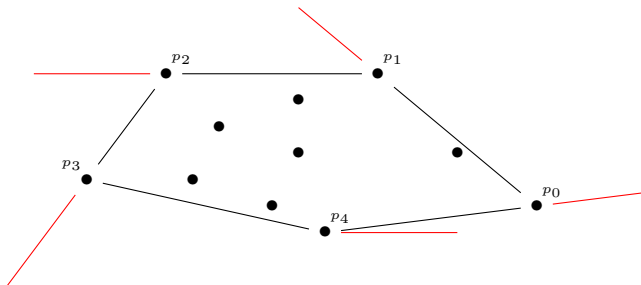
**If**  $p_i \neq p$  **and**  $\overrightarrow{pp_i} \prec \overrightarrow{po}$  w.r.t.  $\overrightarrow{qp}$  **then**  $o \leftarrow p_i$ ;

Insérer( $o$ );

$q \leftarrow p$ ;  $p \leftarrow o$ ;

**Until**  $o = r$ ;

# Illustration et complexité



## Complexité.

La complexité de l'algorithme est en  $O(nh)$

où  $h$  est le nombre de sommets de l'enveloppe convexe.

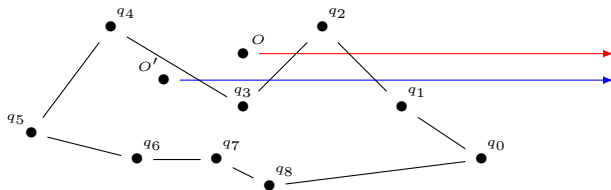
Si  $h = o(\log(n))$  alors cet algorithme plus efficace que les précédents.

# Appartenance à un polygone simple

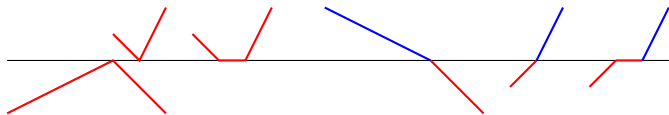
Soit  $O$  un point. On considère la demi-droite horizontale  $\overrightarrow{Ox}$ .

On compte le nombre d'intersections avec les segments du polygone.

$O$  appartient au polygone si ce nombre est impair.



Cas dégénérés.





# Appartenance à un polygone convexe

On détermine en  $O(1)$  un point intérieur  $O$ .

On choisit  $\overrightarrow{Op_0}$  pour repérer les coordonnées polaires.

**Première étape :** recherche du secteur angulaire de  $P$  en  $O(\log(n))$

**If**  $\overrightarrow{Op_{n-1}} \preceq \overrightarrow{OP}$  **then return**  $n - 1$ ;

$i \leftarrow 0$ ;  $j \leftarrow n - 1$ ;

**While**  $j > i + 1$  **do**

$k \leftarrow \lfloor \frac{i+j}{2} \rfloor$ ;

**If**  $\overrightarrow{OP} \prec \overrightarrow{Op_k}$  **then**  $j \leftarrow k$  **else**  $i \leftarrow k$ ;

**return**  $i$ ;

**Deuxième étape :** test du triangle en  $O(1)$

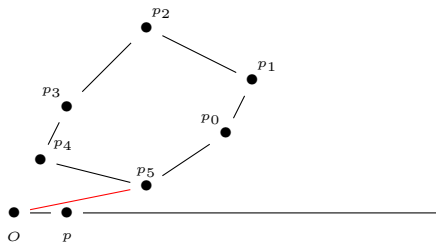
**return**  $P \in \text{Triangle}(O, p_i, p_{i+1 \% n})$ ;

# Point minimal d'un polygone convexe

Soit un polygone convexe  $\mathcal{P} = (p_0, \dots, p_{n-1})$  contenu dans :

- ▶ un demi-plan délimité par la droite portant le segment  $Op$
- ▶ privé de la demi-droite  $\{p' \mid \exists \lambda \geq 0 \overrightarrow{Op'} = \lambda \overrightarrow{Op}\}$ .

On considère l'ordre polaire relatif à  $O$  et  $\overrightarrow{Op}$ .

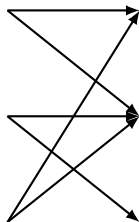


Soit  $p_i$  le point minimal et  $p_j$  le point maximal. Alors :

- ▶  $\overrightarrow{Op_i} \prec \overrightarrow{Op_{i+1 \% n}} \prec \overrightarrow{Op_{i+2 \% n}} \prec \dots \prec \overrightarrow{Op_j}$
- ▶  $\overrightarrow{Op_i} \prec \overrightarrow{Op_{i-1 \% n}} \prec \overrightarrow{Op_{i-2 \% n}} \prec \dots \prec \overrightarrow{Op_j}$

# Réduction d'intervalles

Découpage d'un ou deux intervalles en trois et élimination d'une partie.



## Réduction

- ▶ d'au moins d'un facteur  $\frac{3}{4}$  à chaque étape ;
- ▶ d'au moins d'un facteur  $\frac{1}{2}$  pour deux étapes.

# Recherche du point minimal

$Int \leftarrow [0, n - 1];$

**While**  $|Int| \geq 3$  **do**

//  $Int$  est un intervalle ou l'union de deux intervalles contenant le minimum

$i \leftarrow \text{indfirstpart}(Int);$

$j \leftarrow \text{indsecondpart}(Int);$

$k \leftarrow \text{indthirdpart}(Int);$

$\min \leftarrow \arg \min(\overrightarrow{Op_x} \mid x \in \{i, j, k\});$

**If**  $\min = i$  **then**  $Int \leftarrow Int \setminus \text{secondpart}(Int);$

**If**  $\min = j$  **then**  $Int \leftarrow Int \setminus \text{thirdpart}(Int);$

**If**  $\min = k$  **then**  $Int \leftarrow Int \setminus \text{firstpart}(Int);$

**return**  $\arg \min(\overrightarrow{Op_x} \mid x \in Int);$

A chaque itération le cardinal de  $Int$  est diminué d'au moins un quart  
d'où une complexité en  $O(\log(n))$ .

# Un algorithme incomplet à la Jarvis (1)

## Première étape

On choisit un paramètre  $m \leq n$  *supposé être* une borne supérieure du nombre de points de l'enveloppe convexe.

On partitionne  $\mathcal{P}$  en  $\lceil n/m \rceil$  sous-ensembles d'au plus  $m$  points.

On calcule  $\{\mathcal{P}_i\}_{i \leq \lceil n/m \rceil}$  les enveloppes convexes de ces sous-ensembles en  $O(n \log(m))$ .

## Deuxième étape

On applique une variante de l'algorithme de Jarvis qui :

- ▶ échoue si l'enveloppe convexe contient plus de  $m$  points ;
- ▶ recherche le nouveau point à insérer comme « minimum » des points minimaux des  $\mathcal{P}_i$  ;
- ▶ et par conséquent opère en  $O(n \log(m))$ .

# Un algorithme incomplet à la Jarvis (2)

Soit  $r \in \mathcal{C}$  minimal pour l'ordre lexicographique des coordonnées  $(y, -x)$ .

Insérer( $r$ );

**For**  $i$  **from** 1 **to**  $\lceil n/m \rceil$  **do**

**If**  $s \leftarrow \text{Minpoint}(\mathcal{P}_i, r, \vec{r\hat{x}})$ ;

**If**  $i = 1$  **or**  $\vec{r\hat{s}} \prec \vec{r\hat{o}}$  w.r.t.  $\vec{r\hat{x}}$  **then**  $p \leftarrow s$ ;

Insérer( $p$ );

$q \leftarrow r$ ;

**For**  $k$  **from** 1 **to**  $m$  **do**

**For**  $i$  **from** 1 **to**  $\lceil n/m \rceil$  **do**

**If**  $s \leftarrow \text{Minpoint}(\mathcal{P}_i, p, \vec{q\hat{p}})$ ;

**If**  $i = 1$  **or**  $\vec{p\hat{s}} \prec \vec{p\hat{o}}$  w.r.t.  $\vec{q\hat{p}}$  **then**  $o \leftarrow s$ ;

**If**  $o = r$  **then return**  $\top$  **else** Insérer( $o$ );

$q \leftarrow p$ ;  $p \leftarrow o$ ;

**return**  $\perp$ ;

**Observation.** Si l'algorithme ne s'arrêtait pas après  $m$  itérations, sa complexité serait en  $O(n(\frac{h}{m} + 1) \log(m))$ .

# L'algorithme de Chan

L'algorithme de Chan essaye des valeurs de  $m$  très rapidement croissantes.

```
For  $i$  from 1 to  $\lceil \log(\log(n)) \rceil$  do  
   $m \leftarrow \min(2^{2^i}, n)$  ;  
  If  $\text{IncJarvis}(\mathcal{P}, m) = \top$  then Return ;
```

## Analyse de complexité

Soit  $h$  le nombre de points de l'enveloppe convexe.

L'algorithme s'arrête après  $\lceil \log(\log(h)) \rceil$  itérations.

La  $i$ ème itération s'exécute en  $O(n2^i)$ .

L'algorithme s'exécute donc en  $O(n2^{\lceil \log(\log(h)) \rceil + 1}) = O(n \log(h))$ .