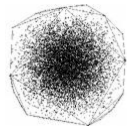
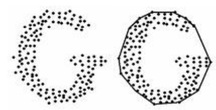


# 5 . Invólucros Convexos no Plano

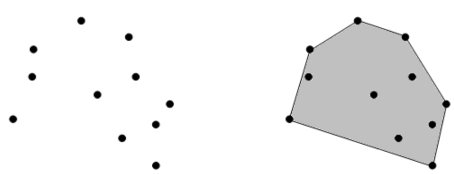


**António Leslie Bajuelos**
Departamento de Matemática
Universidade de Aveiro

**Mestrado em Matemática e Aplicações**

## Problema: uma primeira abordagem…

- **Definição do Problema:**
  - Dado: um conjunto finito *S* de pontos
  - Encontrar: o invólucro convexo dos pontos em *S*



**O Invólucro Convexo de um conjunto $S \subseteq \mathbb{R}^d$ é o menor conjunto convexo de $\mathbb{R}^d$ que contém $S$**

2

## Apontamentos históricos

- Segundo O'Rourke, o problema do invólucro convexo terá motivado o primeiro artigo publicado na área de Geometria Computacional

- Os primeiros algoritmos, desenvolvidos durante os anos 60, têm complexidade $O(n^2)$ ou superior

- Uma boa motivação para continuar a trabalhar…
  - No final dos anos 60, uma aplicação da Bell Labs necessitava de computar o invólucro convexo de aproximadamente 10.000 pontos no plano e os algoritmos de complexidade $O(n^2)$ foram considerados muito lentos

3

## Apontamentos históricos

- Tendo essa aplicação como motivação, Graham desenvolveu em 1972 o primeiro algoritmo de complexidade de tempo $O(n \log n)$

- Actualmente, este problema continua a despertar interesse dos investigadores. Algumas aplicações:
  - Robótica e animação por computador: detecção de colisões
  - Na resolução de outros problemas da Geometria Computacional, como o reconhecimento de padrões, o problema do par mais distante,……

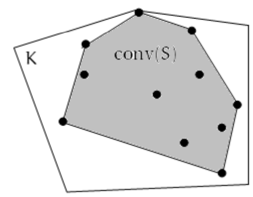
4

## O que ó um invólucro convexo?

- Um conjunto $S \subseteq \mathbb{R}^d$ é convexo quando, para quaisquer pontos *x*, $y \in$ S, o segmento de recta *xy* está totalmente contido em *S*

Conjuntos Convexos — Conjuntos não Convexos

- O **Invólucro Convexo** de um conjunto $S \subseteq \mathbb{R}^d$ é o <u>menor conjunto convexo de $\mathbb{R}^d$ que contém *S*</u> e denota-se por **conv(*S*)**, i.e.: verifica as seguintes propriedades:
  - conv(S) é convexo
  - $S \subseteq$ conv(S)
  - K convexo, $S \subseteq K \Rightarrow$ conv(S) $\subseteq$ K




5

## Resultados Preliminares

- Consequência imediata da definição:
  - **conv(*S*)** é a intersecção de todos os conjuntos convexos que contêm *S*, i.e.:

$$conv(S) = \bigcap_{\substack{K \text{ convexo} \\ S \subseteq K}} K$$

6

## Resultados Preliminares

- Uma caracterização mais concreta usa a noção de combinação convexa:
  - Um ponto $p$ é uma combinação convexa dos pontos $p_1, \ldots, p_n$ se: $p = \lambda_1 p_1 + \lambda_2 p_2 + \ldots + \lambda_n p_n$, onde $\lambda_i \in \mathbb{R}$, $\lambda_i \geq 0$ e $\sum \lambda_i = 1$
  - O **Invólucro Convexo** de $S = \{p_1, p_2, \ldots, p_n\} \subseteq \mathbb{R}^d$ é o conjunto de todos os pontos que são combinação convexa dos pontos em $S$, *i.e.:*

    $\mathrm{conv}(S) = \{\lambda_1 p_1 + \lambda_2 p_2 + \ldots + \lambda_n p_n,\ p_i \in S,\ \lambda_i \in \mathbb{R},\ \lambda_i \geq 0,\ \sum \lambda_i = 1\}$

7

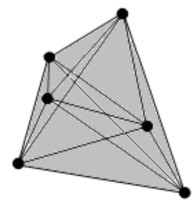
## Resultados Preliminares

- Outros resultados conhecidos:
  - O **Invólucro Convexo** de $S \subseteq \mathbb{R}^d$ é o conjunto de todos os pontos que são combinação convexa de d+1 pontos de $S$ (Teorema de Carathéodory)
  - Como consequência, o **Invólucro Convexo** de $S \subseteq \mathbb{R}^2$ (plano) é o conjunto de combinações convexas dos seus subconjuntos de 3 pontos (se os pontos forem não colineares, conv($S$) é um triângulo)
  - O **Invólucro Convexo** de $S \subseteq \mathbb{R}^d$ é a intersecção de todos os semi-espaços que contêm $S$ (no plano, um semi-espaço é um semiplano)

8

## Resultados Preliminares

- No plano ($\mathbb{R}^2$), sabemos ainda que:
  - ☐ O **Invólucro Convexo** de $S \subseteq \mathbb{R}^2$ é um polígono convexo cujos vértices são pontos de $S$
  - ☐ O **Invólucro Convexo** de $S \subseteq \mathbb{R}^2$ é o menor polígono convexo $P$ que contém $S$, i.e.: $\nexists$ $P'$ convexo: $P \supset P' \supseteq S$
  - ☐ O **Invólucro Convexo** de $S \subseteq \mathbb{R}^2$ é o polígono convexo $P$ com menor área tal que $P \supseteq S$
  - ☐ O **Invólucro Convexo** de $S \subseteq \mathbb{R}^2$ é a união de todos os triângulos determinados por pontos em $S$



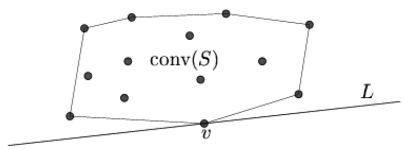


## Problema: uma segunda abordagem (mais precisa)

- Definição do Problema (em $\mathbb{R}^2$):
  - ☐ Dados de Entrada: um conjunto finito $S$ de pontos no plano
  - ☐ Dados de Saída: o conjunto de pontos (vértices) extremos <u>ou</u> o conjunto de segmentos (arestas) extremos que definem o invólucro convexo conv($S$) dos pontos em $S$

# conv($S$) vs pontos extremos de $S$

- Um ponto $p$ é **ponto extremo** de um conjunto S de pontos no plano se $p$ não pode ser escrito como $\lambda x + (1 - \lambda)y$ (combinação convexa), para $x, y \in S$ e $\lambda > 0$. Então:
  - os *pontos extremos* de um conjunto $S$ de pontos no plano são os pontos do conv($S$) que não estão no interior de um segmento contido no conv($S$)
  - os *pontos extremos* de $S$ são os vértices do conv($S$)
  - um ponto $v$ é um ponto extremo de um conjunto $S$ sse existe uma recta cuja intersecção com o conv($S$) é $v$

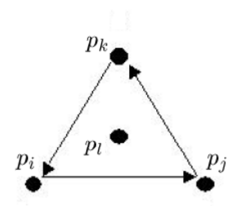



11

# Algoritmos em $\mathbb{R}^2$: ingénuos

- **Lema:** Um ponto $p \in S$ é um ponto não extremo de $S$ sse $p$ pertence ao interior de algum triângulo cujos vértices são pontos de $S$, distintos de $p$
- Algoritmo dos **Pontos-Não-Extremos (CH1)**
  - Dados de Entrada: um conjunto finito $S = \{p_1,\ldots,p_n\}$ de $n$ pontos no plano
  - Dados de Saída: os ponto extremos de $S$ (vértices de conv($S$))
  - Algoritmo:

**for** cada $i$ **do**
  **for** cada $j$, $j \neq i$ **do**
    **for** cada $k$, $k \neq i$, $k \neq j$ **do**
      **for** cada $l$, $l \neq i$, $l \neq j$, $l \neq k$ **do**
        **if** $p_l$ está à esquerda ou sobre $(p_i,p_j)$ e
          $p_l$ está à esquerda ou sobre $(p_j,p_k)$ e
          $p_l$ à esquerda ou sobre $(p_k,p_i)$
        **then** $p_l$ não é extremo

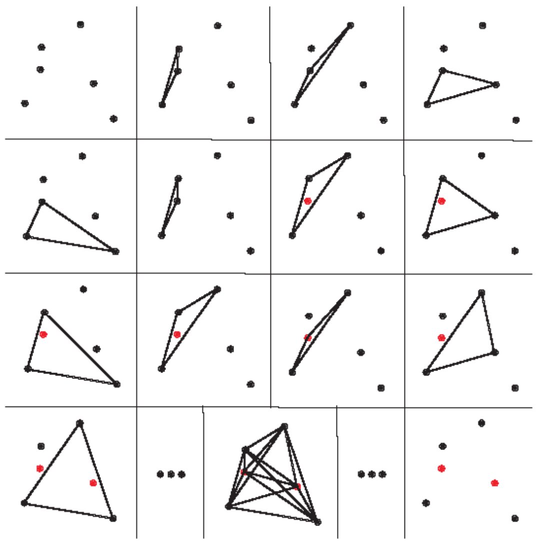



  - **Complexidade: $O(n^4)$**: existem 4 ciclos for encadeados, cada um com $n$ iterações

12

## Algoritmos em $\mathbb{R}^2$: ingénuos

- Exemplo da execução do Algoritmo dos **Pontos-Não-Extremos**



13

## conv(*S*) vs arestas extremas de *S*

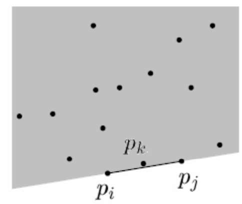
- **Teorema (a):** *pq é uma aresta orientada positivamente de* conv(*S*) *sse todos os pontos de S estão à esquerda de pq ou sobre pq.*
- **Teorema (b):** *As arestas de* conv(*S*) *são definidas por pontos de S e estão nas rectas suporte de S.*
- O algoritmo a seguir pode ser compreendido em duas etapas:
  1. identificar as arestas do conv(*S*) a partir do critério do teorema (a);
  2. representar a fronteira de conv(*S*) a partir das arestas identificadas, tendo em conta o teorema (b).

14

## Algoritmos em $\mathbb{R}^2$ : ingénuos

- Algoritmo das **Arestas-Extremas (CH2)**
  - Dados de Entrada: um conjunto finito $S = \{p_1,\ldots,p_n\}$ de n pontos no plano
  - Dados de Saída: as arestas extremas de $S$ (arestas de conv($S$))
  - Algoritmo:

**for** cada $i$ **do**
  **for** cada $j$, $j \neq i$ **do**
    **for** cada $k$, $k \neq i$, $k \neq j$ **do**
      **if** $p_K$ não está à esquerda ou sobre $(p_i,p_j)$
      **then** $(p_i,p_j)$ não é aresta de conv($S$)

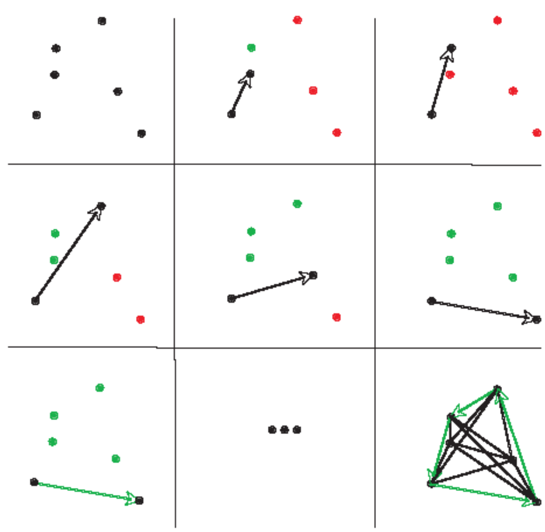



  - **Complexidade: O($n^3$)**: existem 3 ciclos for encadeados, cada um com $n$ iterações

15

## Algoritmos em $\mathbb{R}^2$ : ingénuos

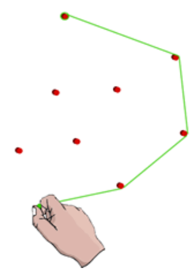
- Exemplo da execução do Algoritmo dos **Arestas-Extremas**



16

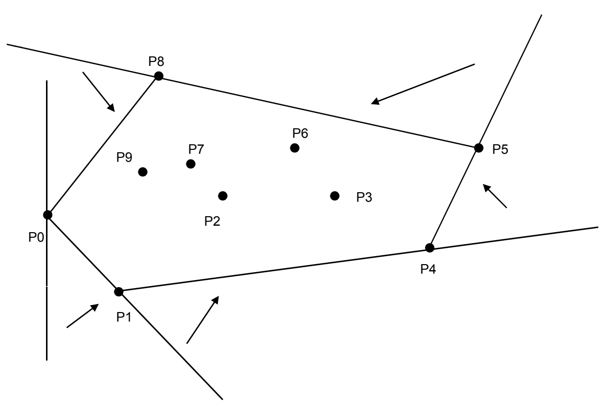
## Algoritmo de Jarvis
(algoritmo do embrulho para presente - **Gift wrapping**)

- O algoritmo de Jarvis baseia-se na seguinte observação:
  - **Se *pq* é uma aresta do conv(*S*) orientada positivamente, então a próxima aresta de conv(*S*) é *qr*, onde *r* é o ponto de S tal que *qr* define o menor ângulo com *pq***
- Este algoritmo simula o enrolar do conjunto *S* por um barbante e pode também ser visto como uma melhoria do *algoritmo das arestas extremas*:
- Ideia central:
  - **utilizar a aresta extrema que acabou de ser encontrada para encontrarmos a próxima aresta.**



17

## Algoritmo de Jarvis
(algoritmo do embrulho para presente - **Gift wrapping**)

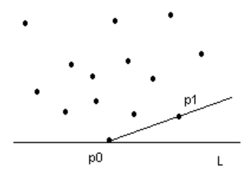



18

# Algoritmo de Jarvis

(algoritmo do embrulho para presente - **Gift wrapping**)

- Ideia central:
  - **utilizar a aresta extrema que acabou de ser encontrada para encontrarmos a próxima aresta.**
- No início não temos nenhuma aresta de conv($S$), então como iniciar o algoritmo?
  - **basta começarmos com um vértice $p_0$ em $S$ de menor $y$-coordenada e encontrarmos o ponto $p_1$ em $S$ com o menor ângulo polar no sentido anti-horário com relação ao ponto $p_0$ e a recta horizontal.**




19

# Algoritmo de Jarvis

(algoritmo do embrulho para presente - **Gift wrapping**)

- Algoritmo de **Jarvis (CH3)**
  - Dados de Entrada: um conjunto finito $S = \{p_1,\ldots,p_n\}$ de $n$ pontos no plano
  - Dados de Saída: as arestas que definem conv($S$), orientadas no sentido positivo
  - Algoritmo:

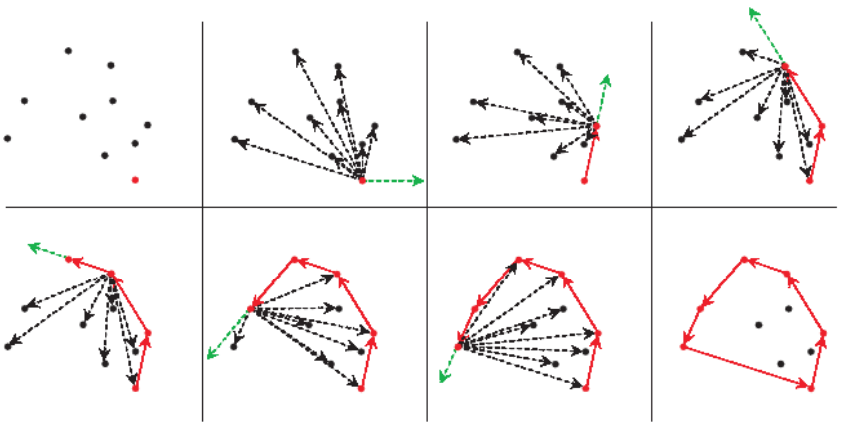
Encontrar o ponto $p_m$ de menor ordenada
Inicializar:
    $i \leftarrow m$
    $v \leftarrow$ vector horizontal que passa por $p_m$ e é orientado
        no sentido positivo
Repeat
    for cada $j$, $j \neq i$ do
        Calcular $\theta_j$ ângulo polar entre $v$ e $p_i p_j$
        Obter $k$ tal que $\theta_k = \min\theta_j$
    Output : $p_i p_k$ como aresta orientada de conv($S$)
    Incrementar: $v \leftarrow p_i p_k$
                $i \leftarrow k$
Until $i = m$

20

## Algoritmo de Jarvis

(algoritmo do embrulho para presente - **Gift wrapping**)

- Exemplo da execução do Algoritmo de **Jarvis**



21

## Algoritmo de Jarvis

(algoritmo do embrulho para presente - **Gift wrapping**)

- Complexidade: ***O*(*nh*)**

$O(n)$ → Encontrar o ponto $p_m$ de menor ordenada

$O(1)$ → Inicializar:
- $i \leftarrow m$
- $v \leftarrow$ vector horizontal que passa por $p_m$ e é orientado no sentido positivo

$O(nh)$:

Repeat

$O(n)$ → for cada $j$, $j \neq i$ do
- Calcular $\theta_j$ ângulo polar entre $v$ e $p_i p_j$
- Obter $k$ tal que $\theta_k = \min \theta_j$

$O(1)$ →
- Output : $p_i p_k$ como aresta orientada de conv($S$)
- Incrementar: $v \leftarrow p_i p_k$
- $i \leftarrow k$

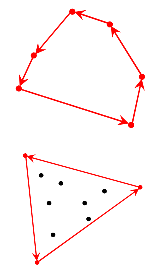
Until $i = m$

22

# Algoritmo de Jarvis

(algoritmo do embrulho para presente - **Gift wrapping**)

- **Observações relevantes:**



1. No pior caso, o algoritmo de **Jarvis** é $O(n^2)$ – corresponde à situação em que todos os pontos de $S$ pertencem ao invólucro convexo
2. No entanto, se o número de arestas $h$ do invólucro é pequeno, digamos log $n$, então o algoritmo de **Jarvis** é $O(n\log n)$ (óptimo???)
3. Em termos de implementação não precisamos calcular ângulo polar algum. Basta usarmos a primitiva ***Left*** para encontrar em cada iteração, o ponto $p_k$ tal que o predicado ***Left***$(p_i, p_k, p_j)$ é verdadeiro para todo $j$ distintos de $i$ e $k$.

23