Computational Geometry

Lecture 9:
Convex Hulls in 3D
or
Mixing More Things

Part I:
Complexity & Visibility

Philipp Kindermann

Winter Semester 2020

Complexity of the Convex Hull

Given set S of n points in $\mathbb{R}^d$,

Complexity of the Convex Hull

Given set S of n points in $\mathbb{R}^d$, what is max. #edges on $\partial\mathrm{CH}(S)$?

Complexity of the Convex Hull

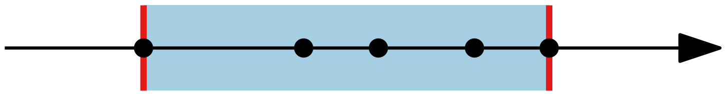
Given set S of n points in $\mathbb{R}^d$, what is max. #edges on $\partial\mathrm{CH}(S)$?

dim	w-c complexity of $\mathrm{CH}(S)$
1	
2	
3	
d	

Complexity of the Convex Hull

Given set S of n points in $\mathbb{R}^d$, what is max. #edges on $\partial\mathrm{CH}(S)$?

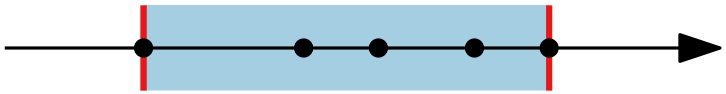
dim	w-c complexity of $\mathrm{CH}(S)$
1	
2	
3	
d	



Complexity of the Convex Hull

Given set S of n points in $\mathbb{R}^d$, what is max. #edges on $\partial\mathrm{CH}(S)$?

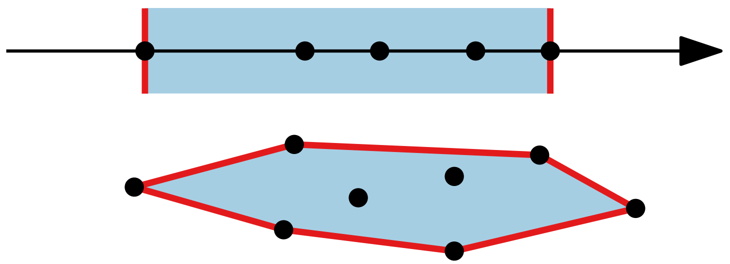
dim	w-c complexity of $\mathrm{CH}(S)$
1	$2 \in \Theta(1)$
2	
3	
d	



Complexity of the Convex Hull

Given set S of n points in $\mathbb{R}^d$, what is max. #edges on $\partial\mathrm{CH}(S)$?

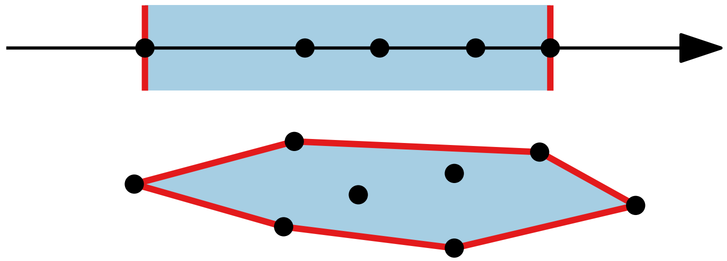
dim	w-c complexity of $\mathrm{CH}(S)$
1	$2 \in \Theta(1)$
2	
3	
d	



Complexity of the Convex Hull

Given set S of n points in $\mathbb{R}^d$, what is max. #edges on $\partial\mathrm{CH}(S)$?

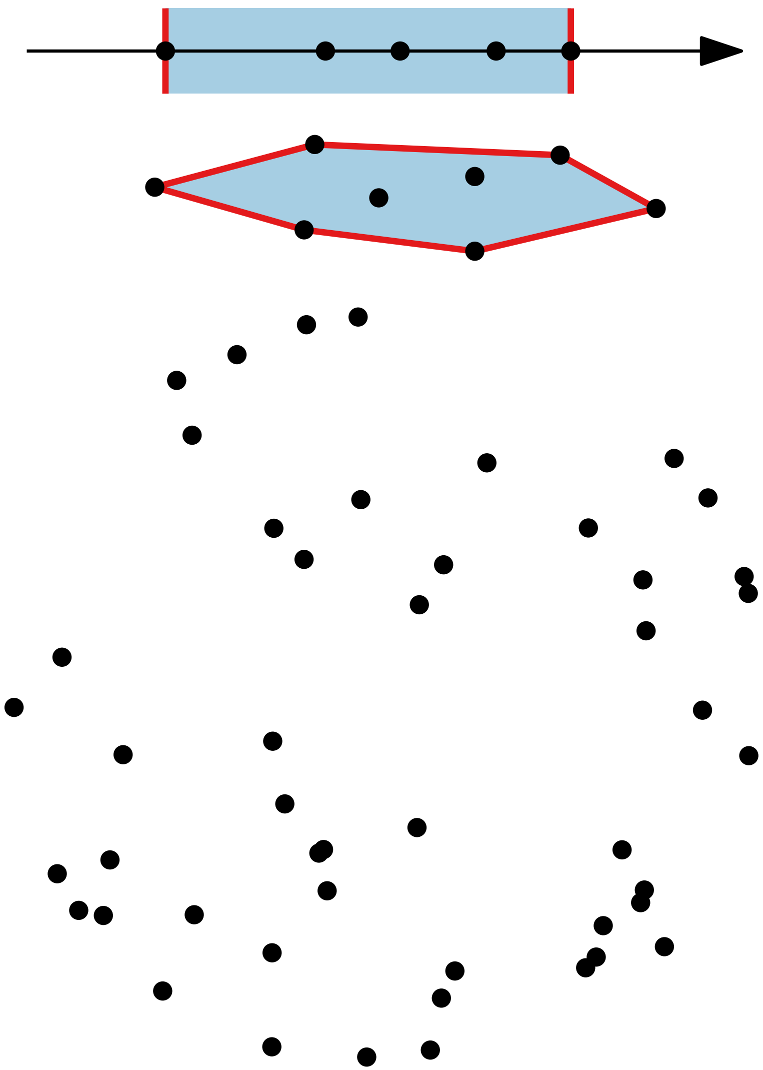
dim	w-c complexity of $\mathrm{CH}(S)$
1	$2 \in \Theta(1)$
2	$n \in \Theta(n)$
3	
d	



Complexity of the Convex Hull

Given set S of n points in $\mathbb{R}^d$, what is max. #edges on $\partial\mathrm{CH}(S)$?

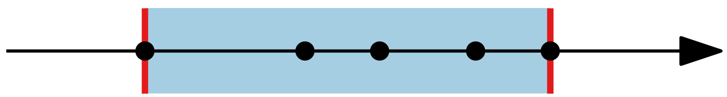
dim	w-c complexity of $\mathrm{CH}(S)$
1	$2 \in \Theta(1)$
2	$n \in \Theta(n)$
3	
d	

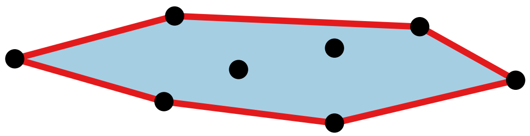


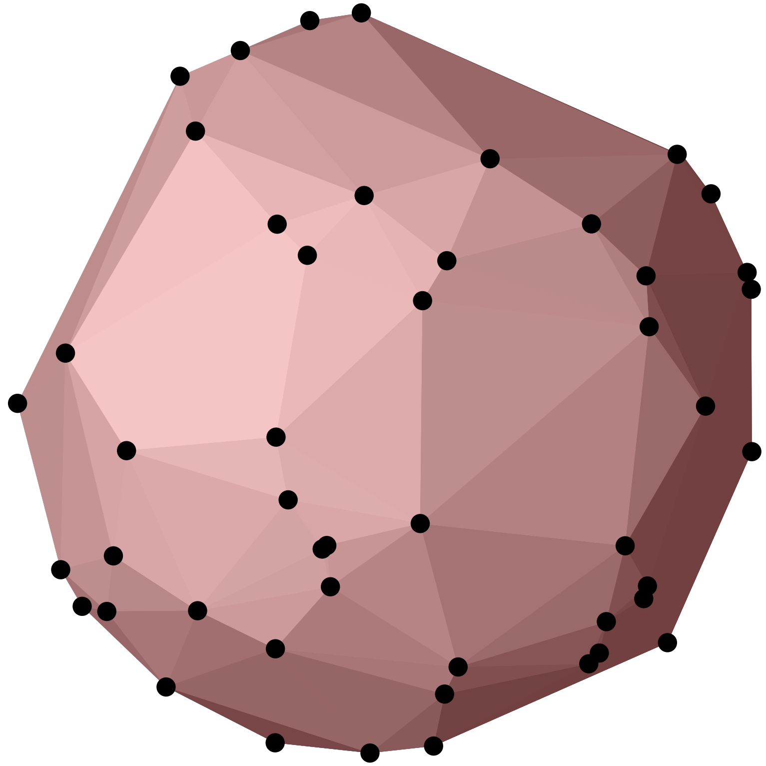
Complexity of the Convex Hull

Given set S of n points in $\mathbb{R}^d$, what is max. #edges on $\partial\mathrm{CH}(S)$?

dim	w-c complexity of $\mathrm{CH}(S)$
1	$2 \in \Theta(1)$
2	$n \in \Theta(n)$
3	
d	



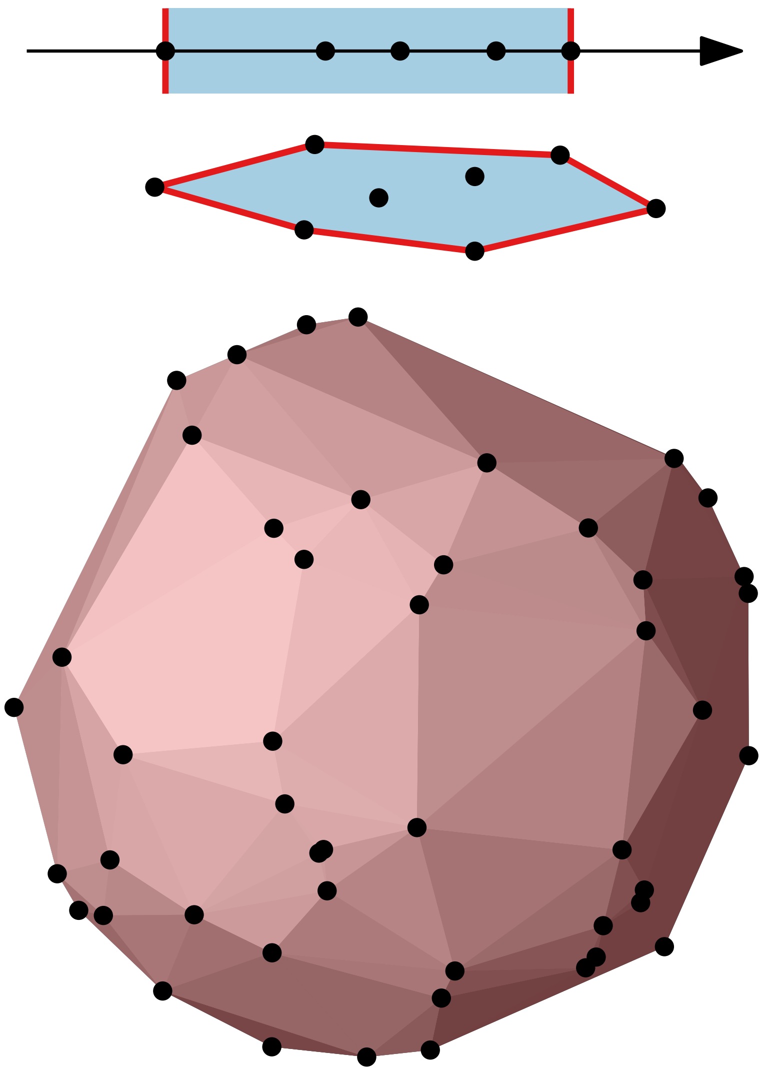




Complexity of the Convex Hull

Given set S of n points in $\mathbb{R}^d$, what is max. #edges on $\partial\mathrm{CH}(S)$?

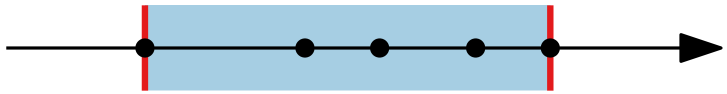
dim	w-c complexity of $\mathrm{CH}(S)$
1	$2 \in \Theta(1)$
2	$n \in \Theta(n)$
3	**Your task!**
d	

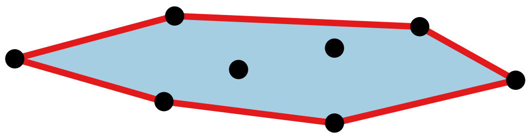


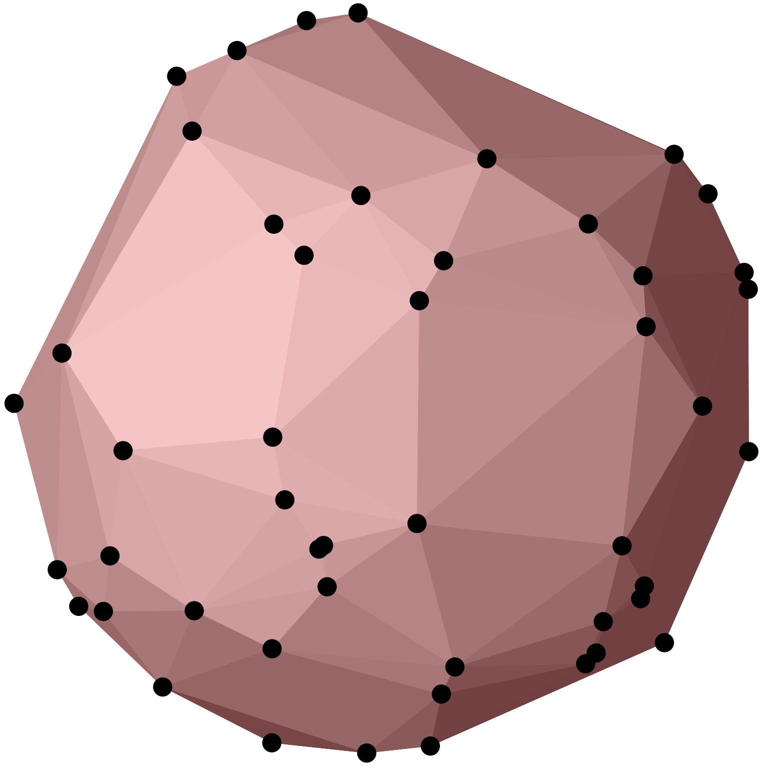
Complexity of the Convex Hull

Given set S of n points in $\mathbb{R}^d$, what is max. #edges on $\partial\mathrm{CH}(S)$?

dim	w-c complexity of $\mathrm{CH}(S)$
1	$2 \in \Theta(1)$
2	$n \in \Theta(n)$
3	$3n - 6 \in \Theta(n)$
d	



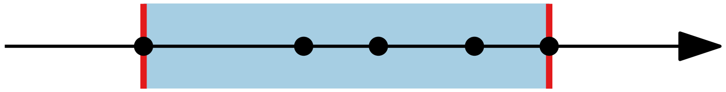
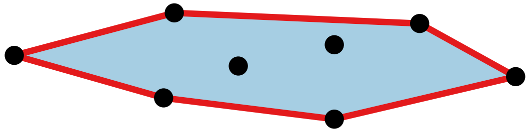
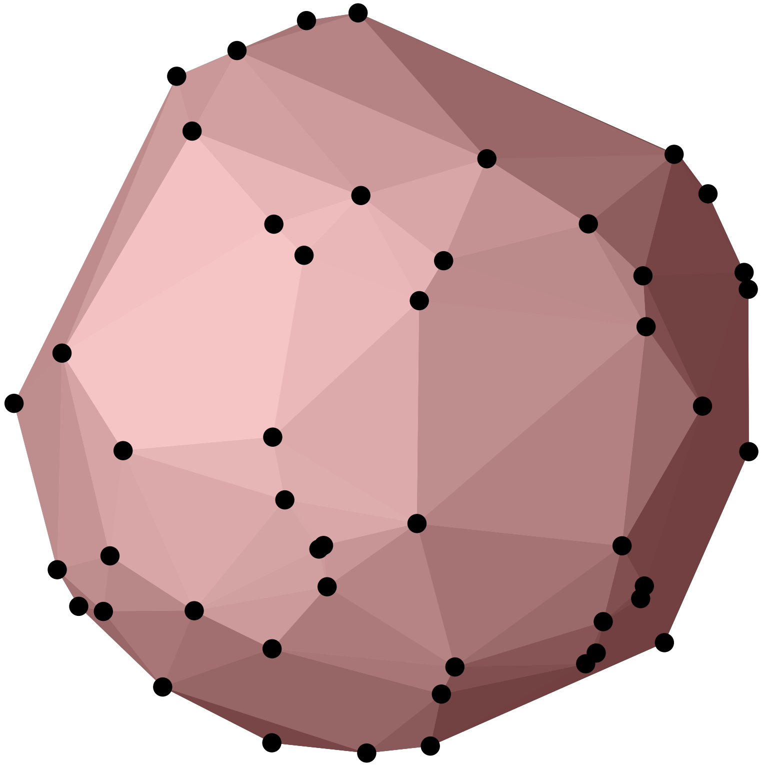




Complexity of the Convex Hull

Given set S of n points in $\mathbb{R}^d$, what is max. #edges on $\partial\mathrm{CH}(S)$?

dim	w-c complexity of $\mathrm{CH}(S)$
1	$2 \in \Theta(1)$
2	$n \in \Theta(n)$
3	$3n - 6 \in \Theta(n)$
d	$\Theta(n^{\lfloor d/2 \rfloor})$

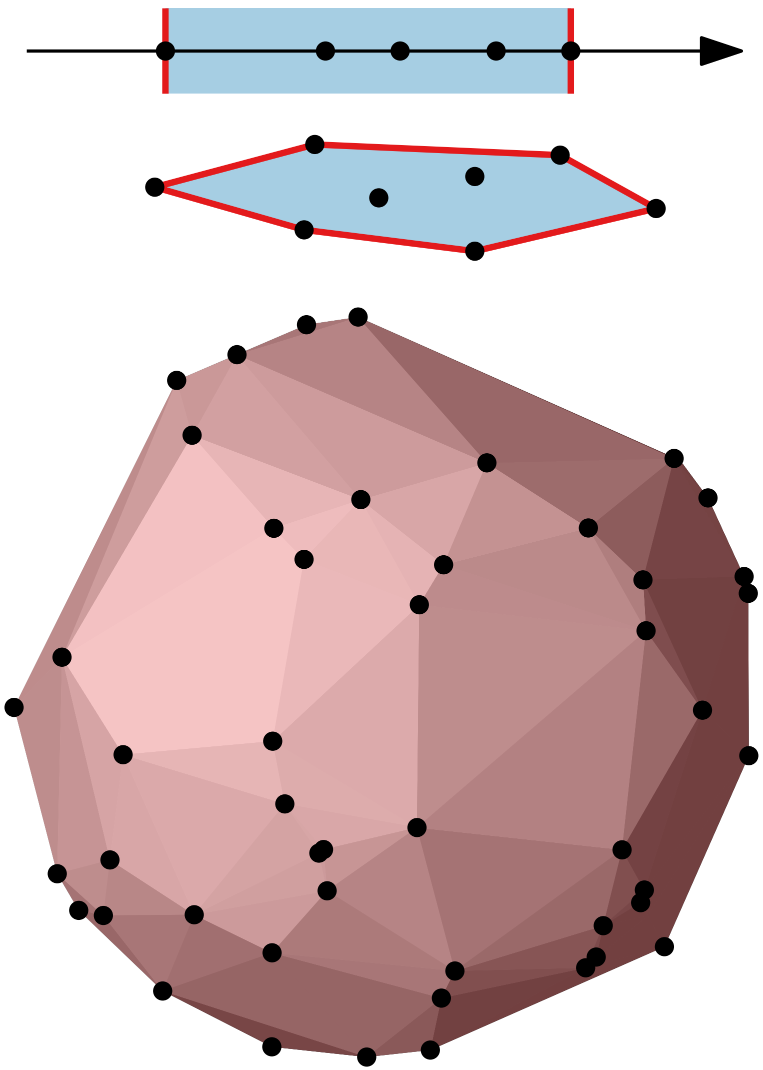




Complexity of the Convex Hull

Given set S of n points in $\mathbb{R}^d$, what is max. #edges on $\partial \mathrm{CH}(S)$?

dim	w-c complexity of $\mathrm{CH}(S)$
1	$2 \in \Theta(1)$
2	$n \in \Theta(n)$
3	$3n - 6 \in \Theta(n)$
d	$\Theta(n^{\lfloor d/2 \rfloor})$

Upper Bound Theorem

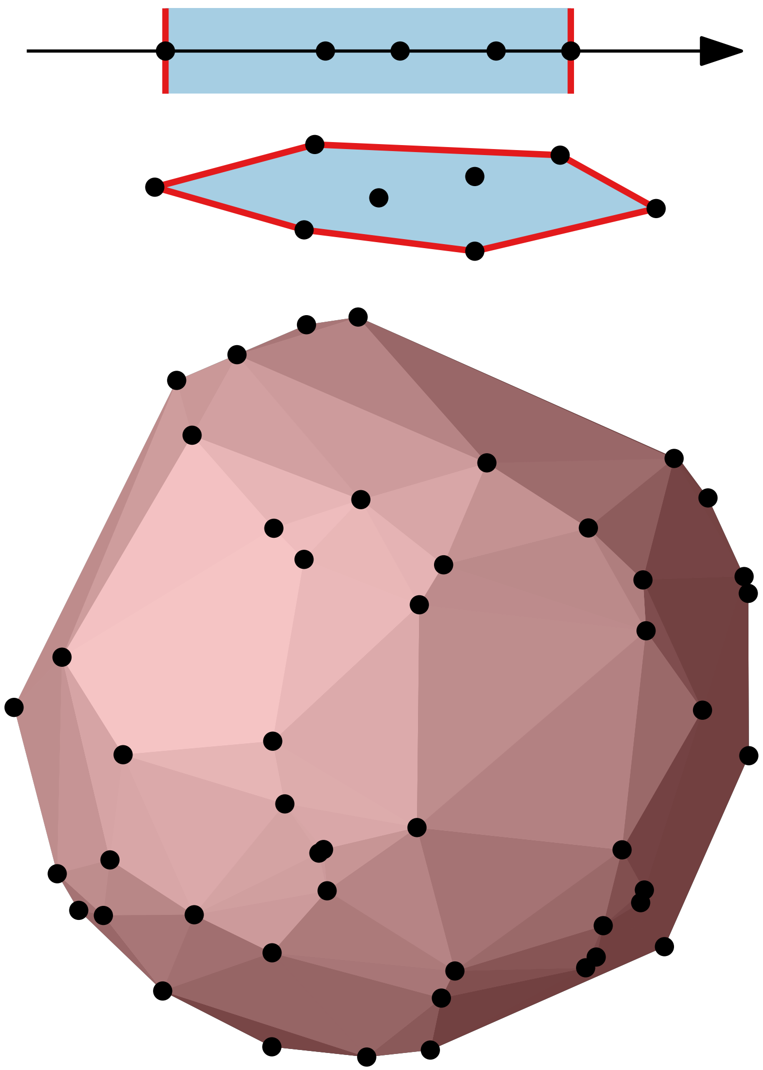


Complexity of the Convex Hull

Given set S of n points in $\mathbb{R}^d$, what is max. #edges on $\partial\mathrm{CH}(S)$?

dim	w-c complexity of $\mathrm{CH}(S)$
1	$2 \in \Theta(1)$
2	$n \in \Theta(n)$
3	$3n - 6 \in \Theta(n)$
d	$\Theta(n^{\lfloor d/2 \rfloor})$

Upper Bound Theorem



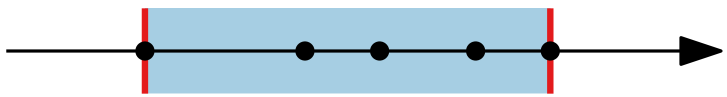
Construction?

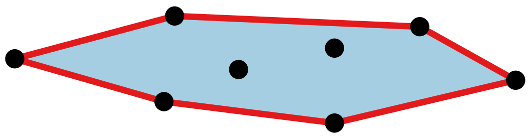
Complexity of the Convex Hull

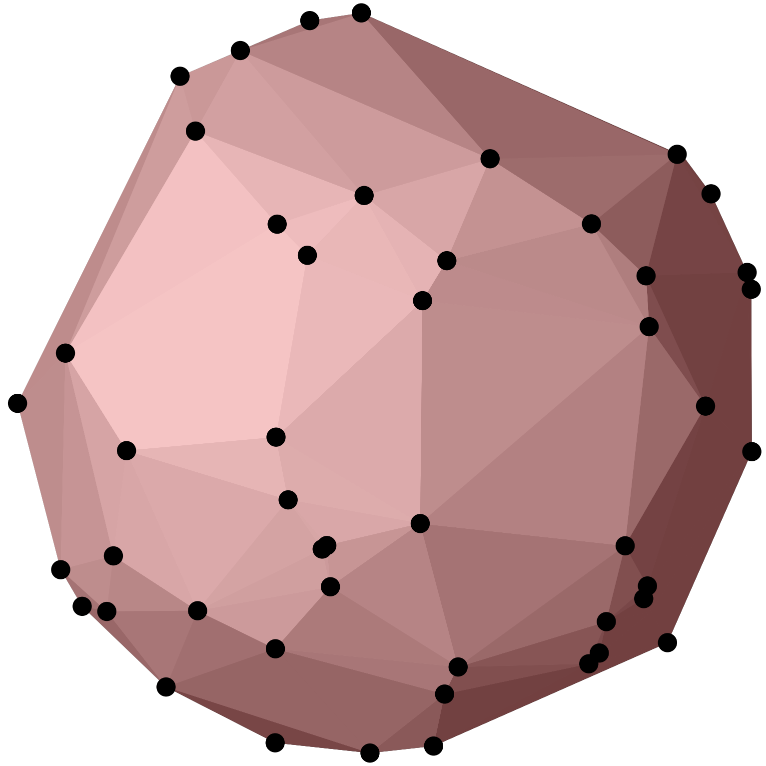
Given set S of n points in $\mathbb{R}^d$, what is max. #edges on $\partial\mathrm{CH}(S)$?

dim	w-c complexity of $\mathrm{CH}(S)$
1	$2 \in \Theta(1)$
2	$n \in \Theta(n)$
3	$3n - 6 \in \Theta(n)$
d	$\Theta(n^{\lfloor d/2 \rfloor})$

Upper Bound Theorem



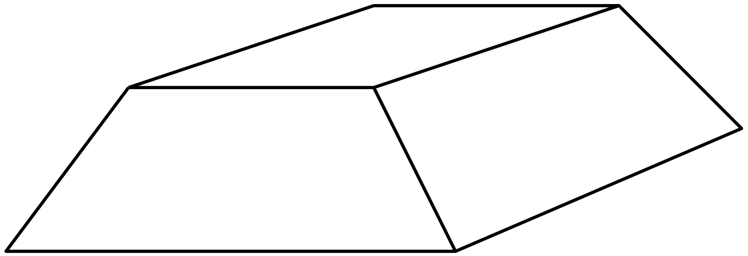




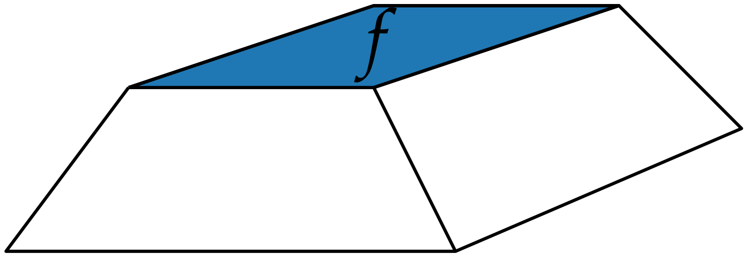
Construction

randomized-incremental!

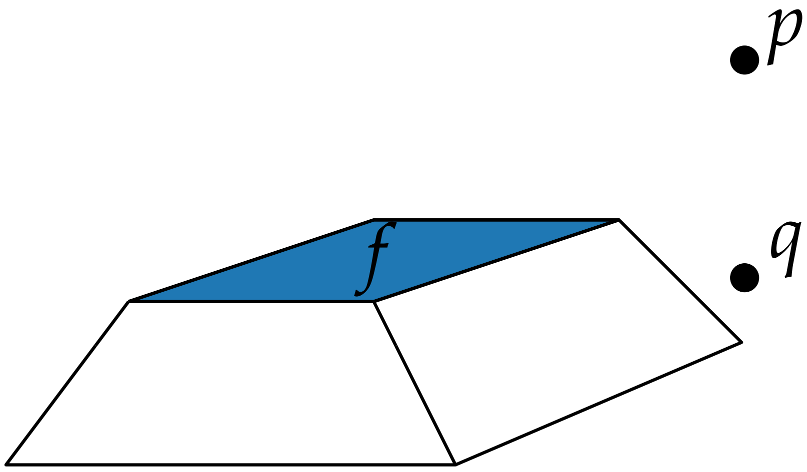
Visibility



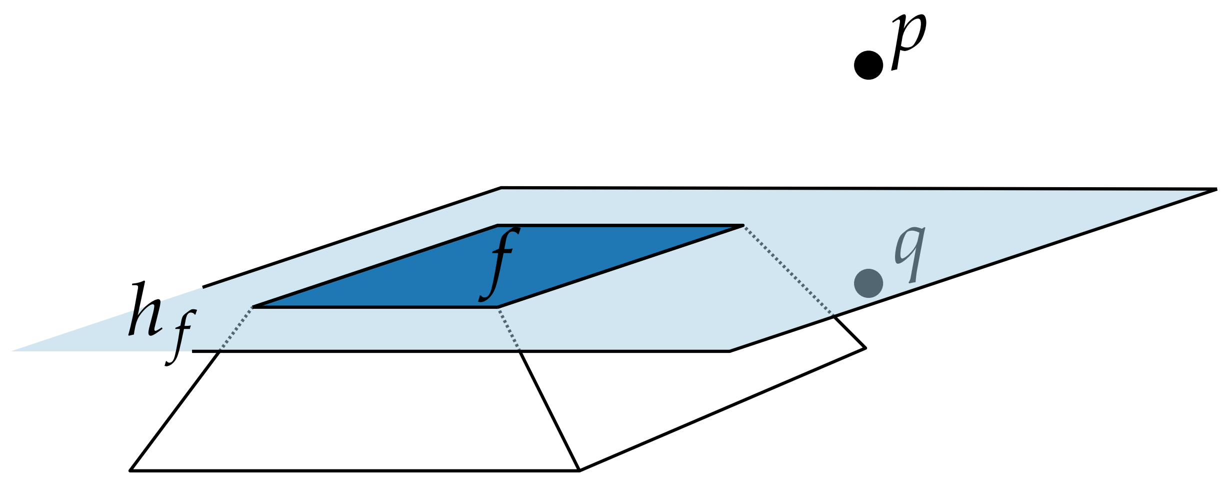
Visibility



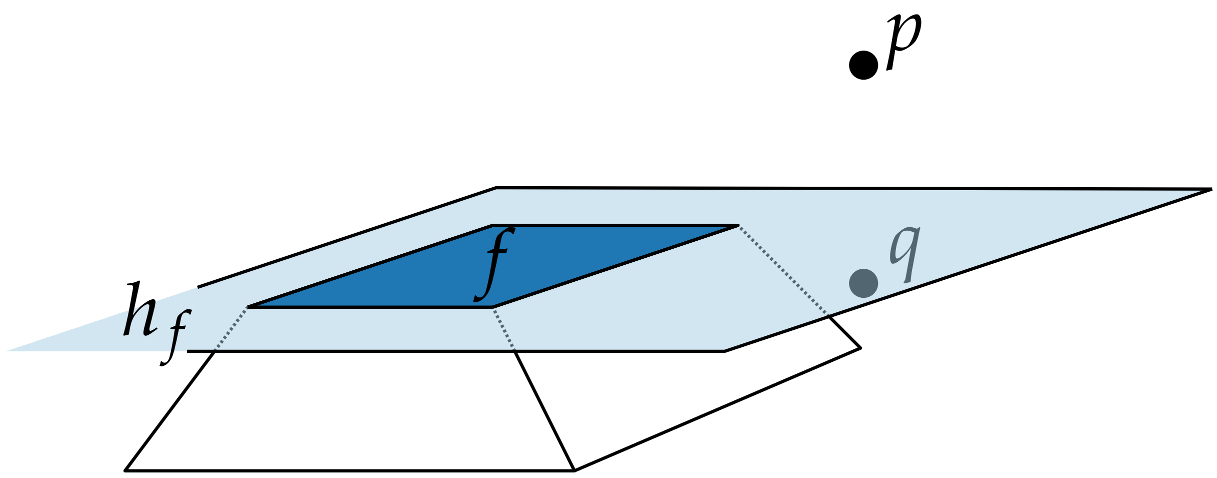
Visibility



Visibility

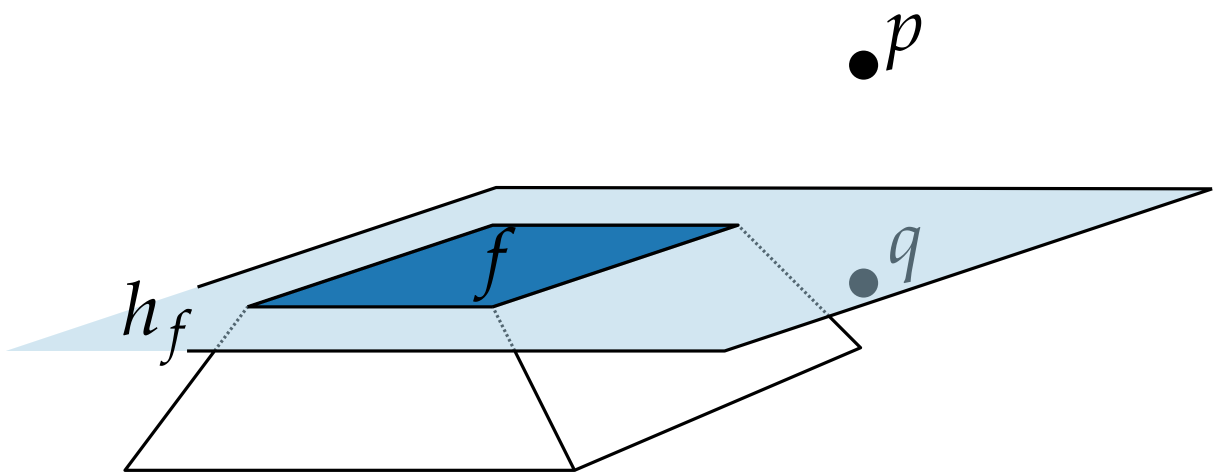


Visibility

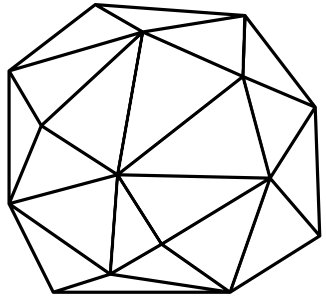


Face f is *visible* from p but not from q.

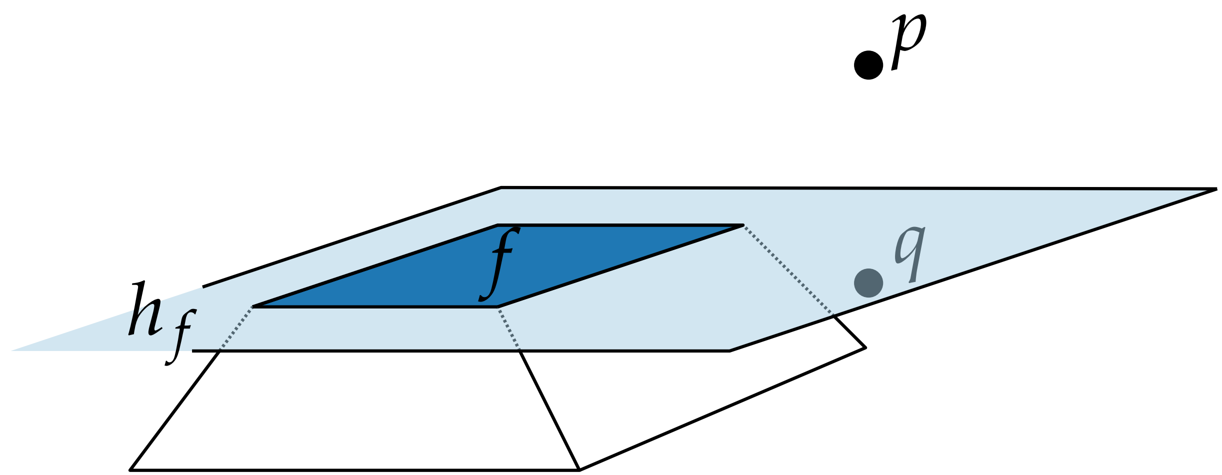
Visibility



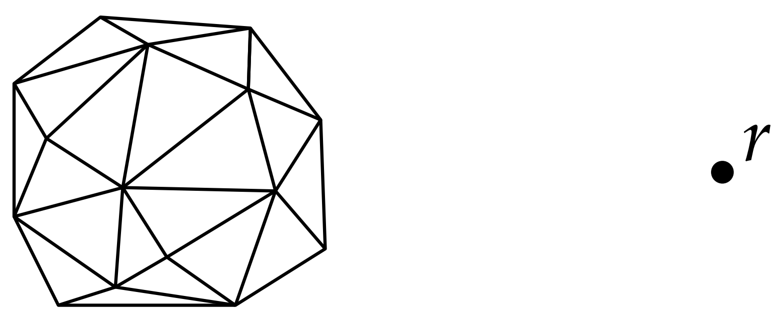
Face f is *visible* from p but not from q.



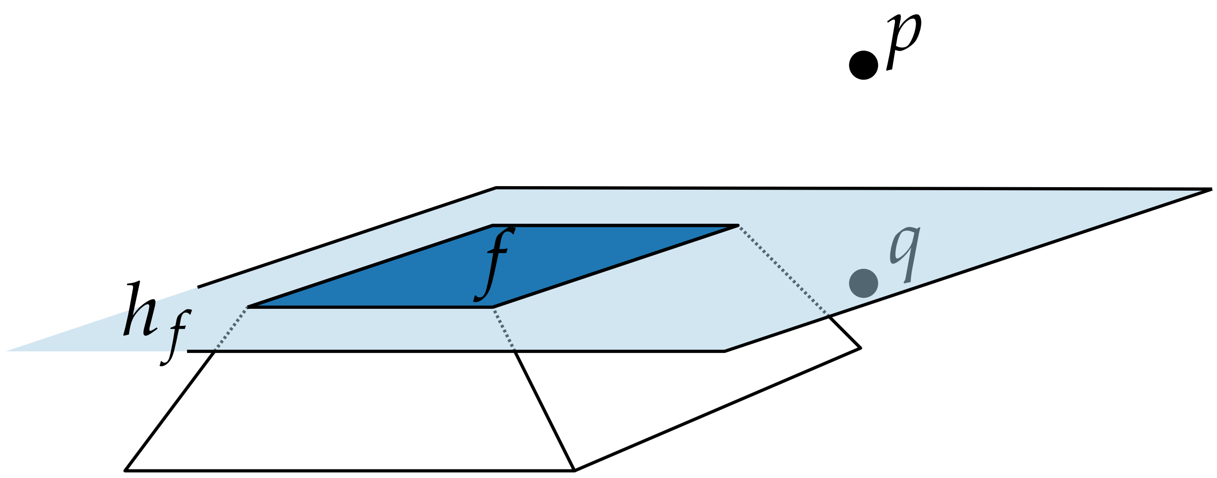
Visibility



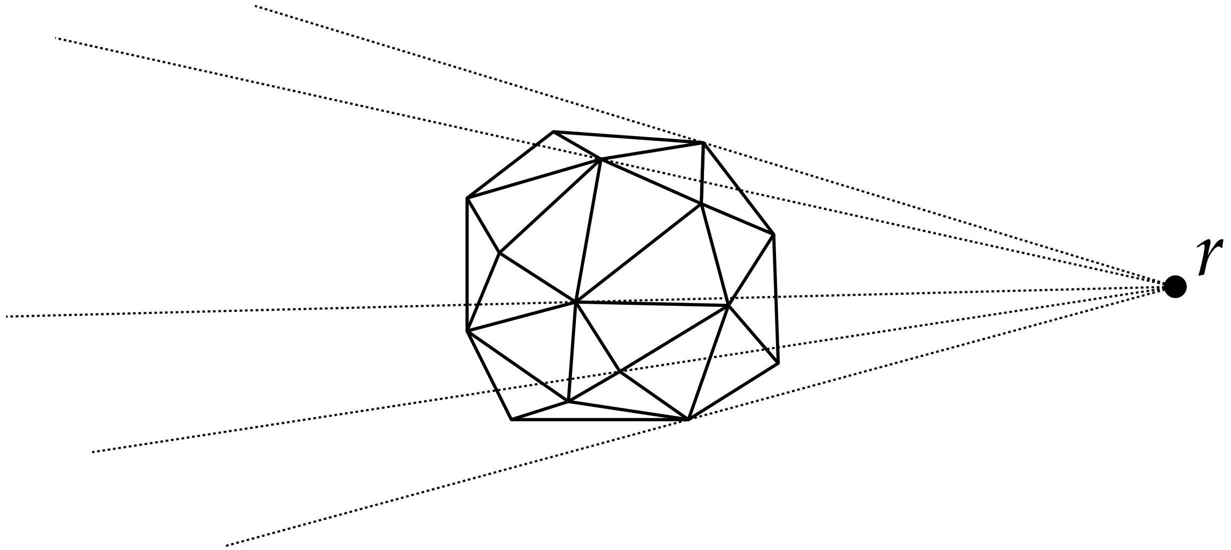
Face f is *visible* from p but not from q.



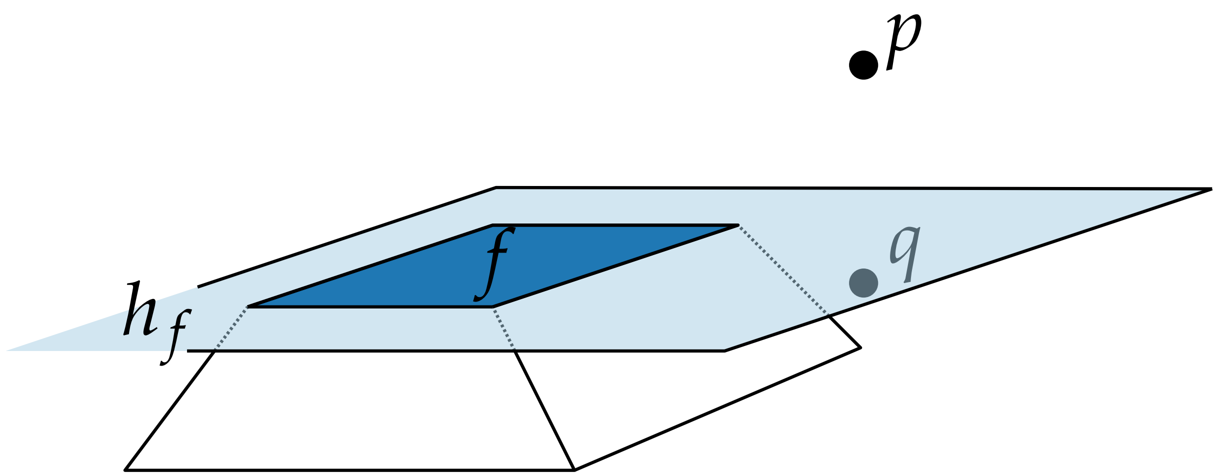
Visibility



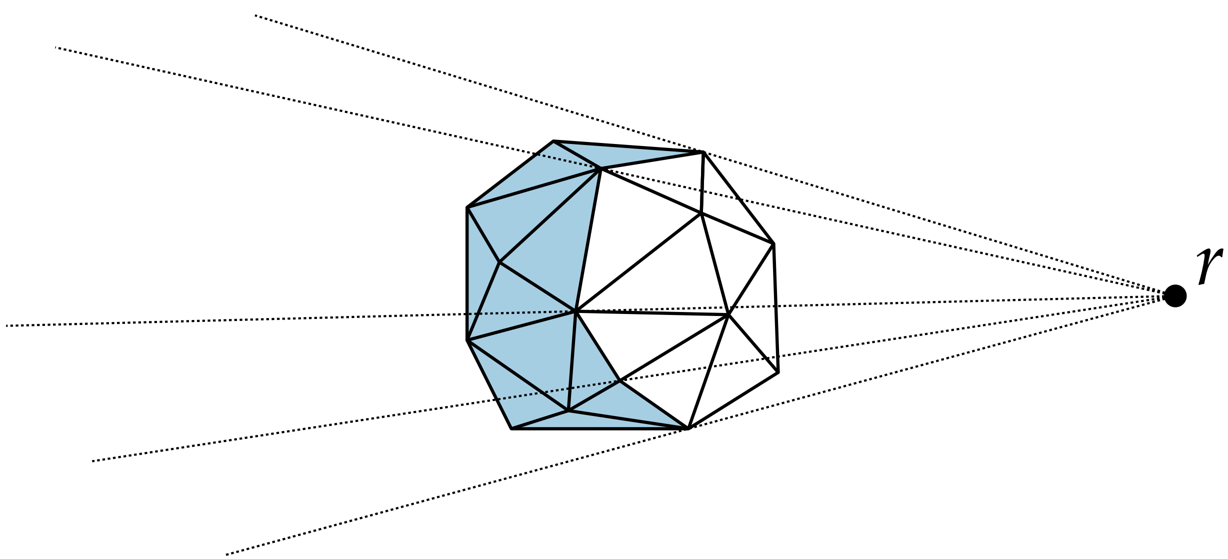
Face f is *visible* from p but not from q.



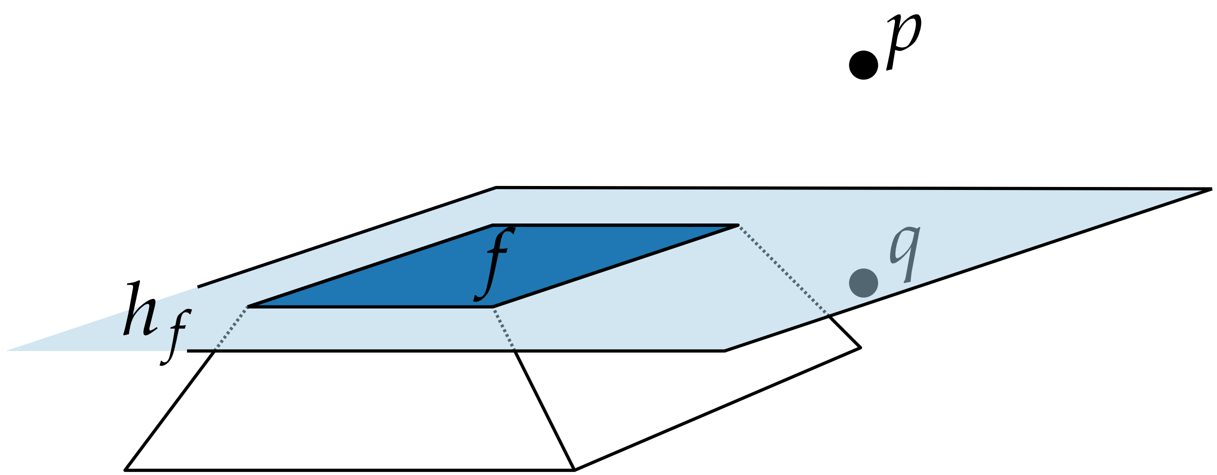
Visibility



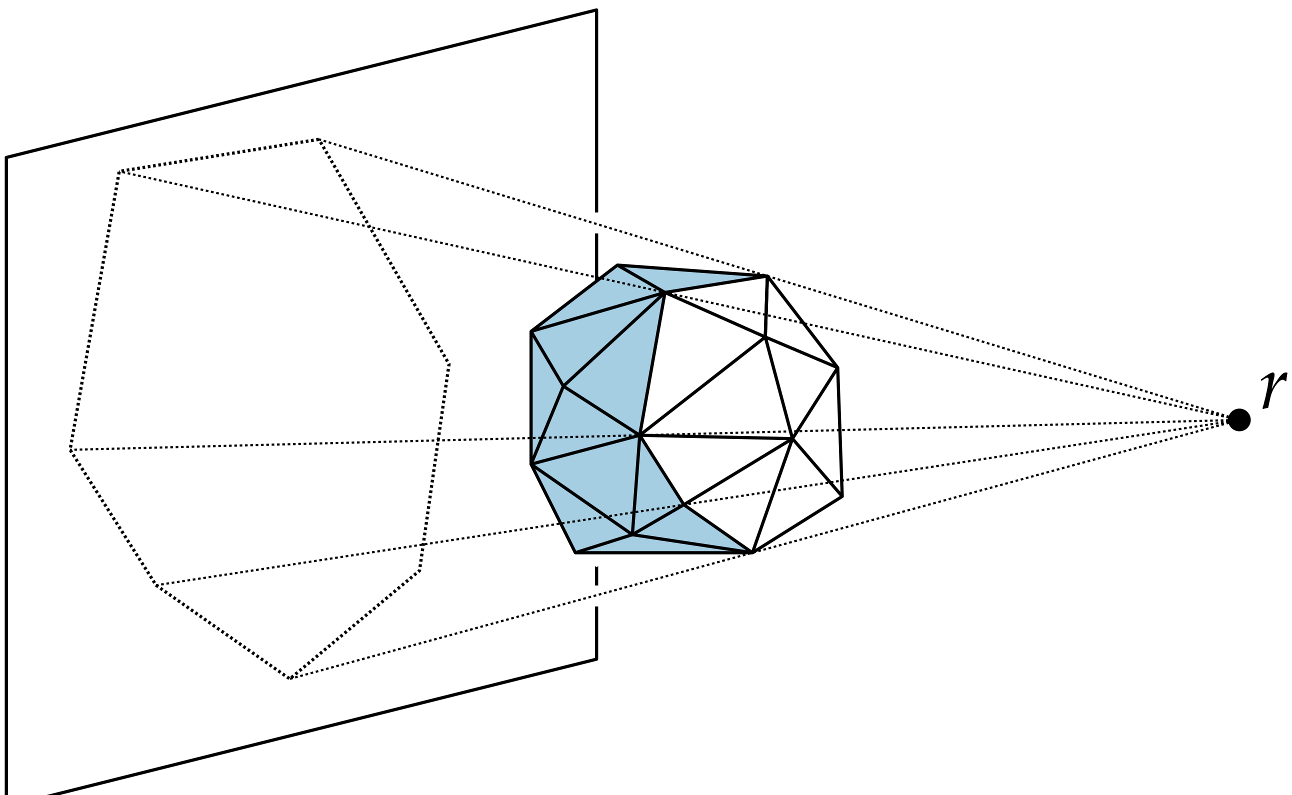
Face f is *visible* from p but not from q.



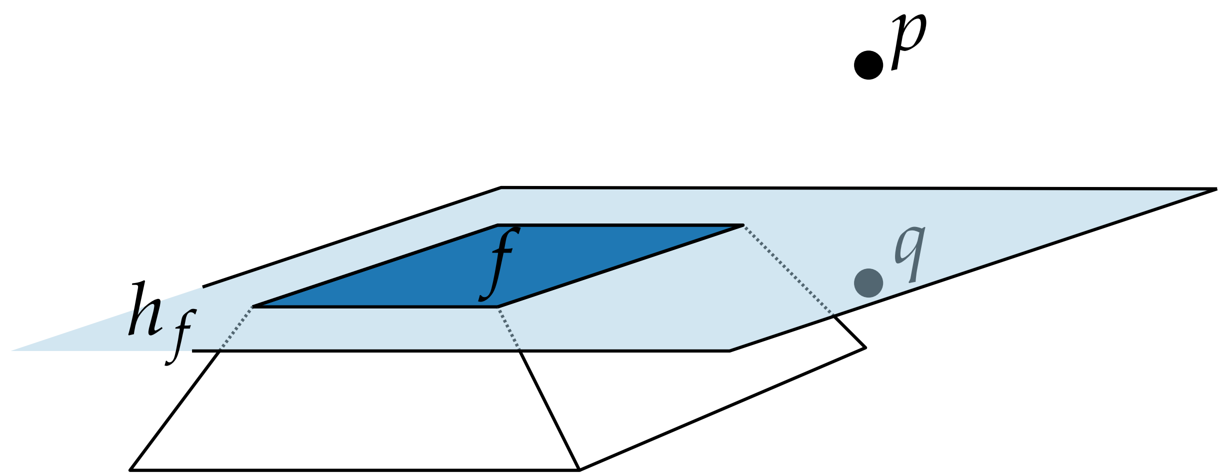
Visibility



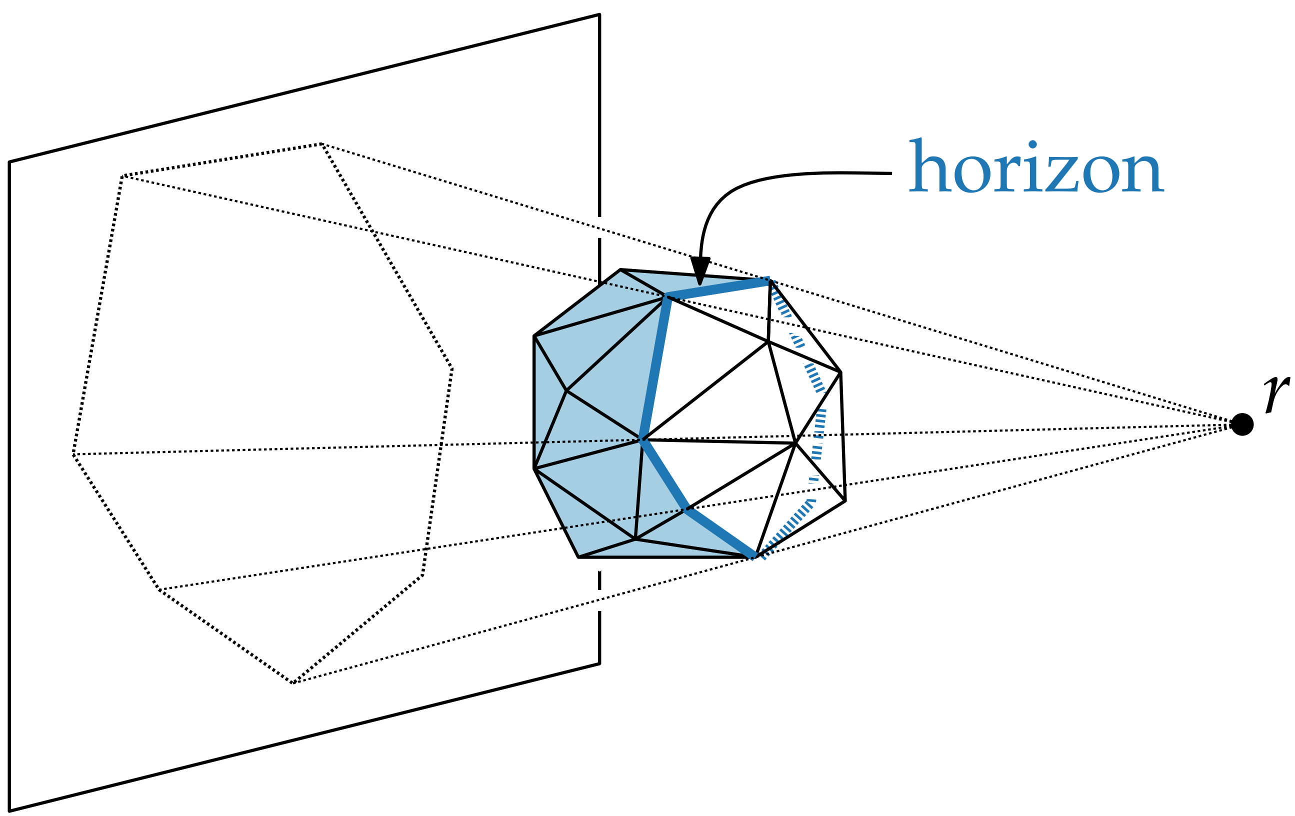
Face f is *visible* from p but not from q.



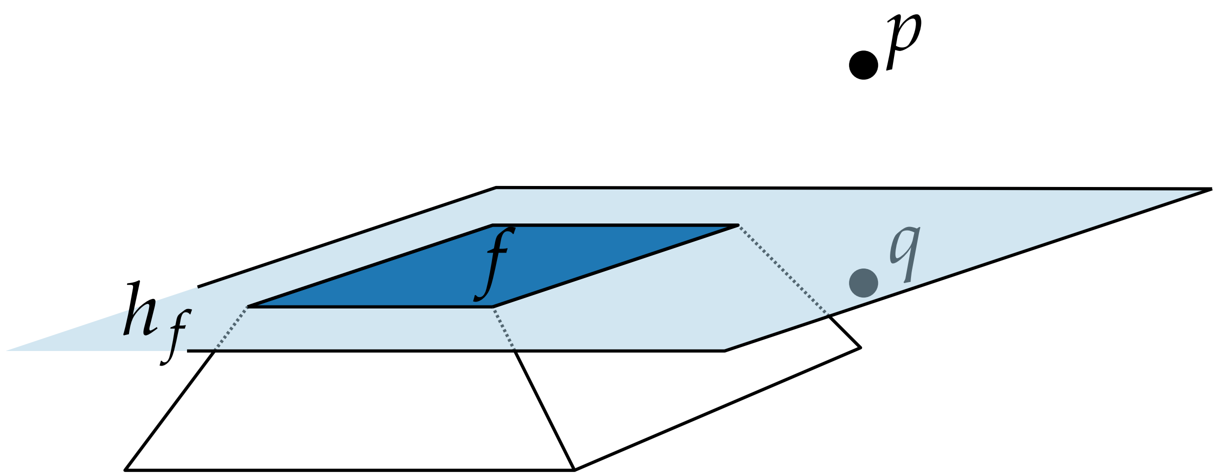
Visibility



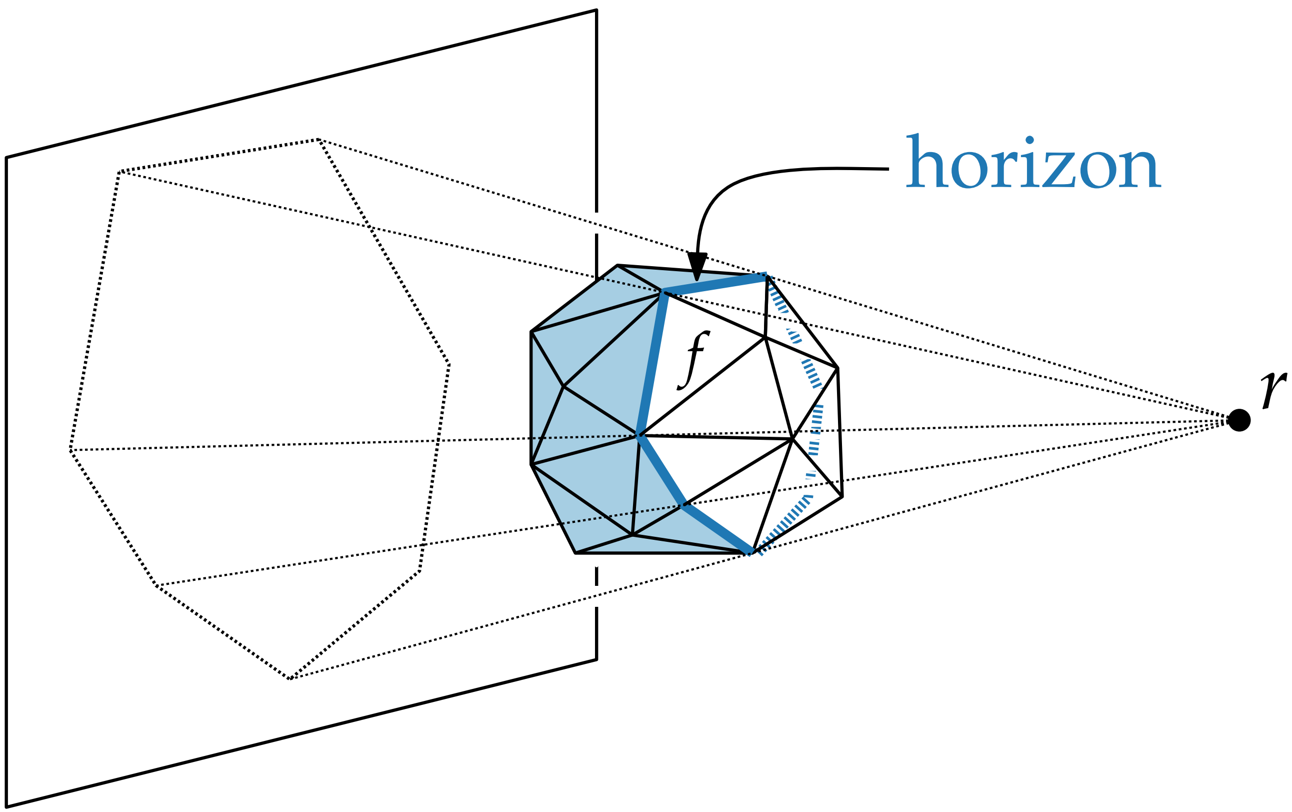
Face f is *visible* from p but not from q.



Visibility

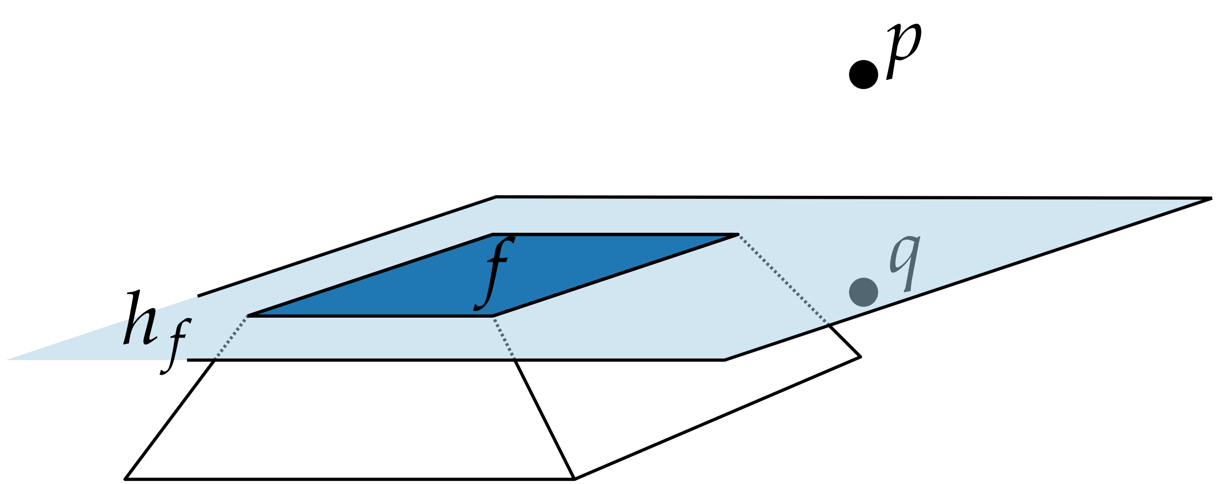


Face f is *visible* from p but not from q.

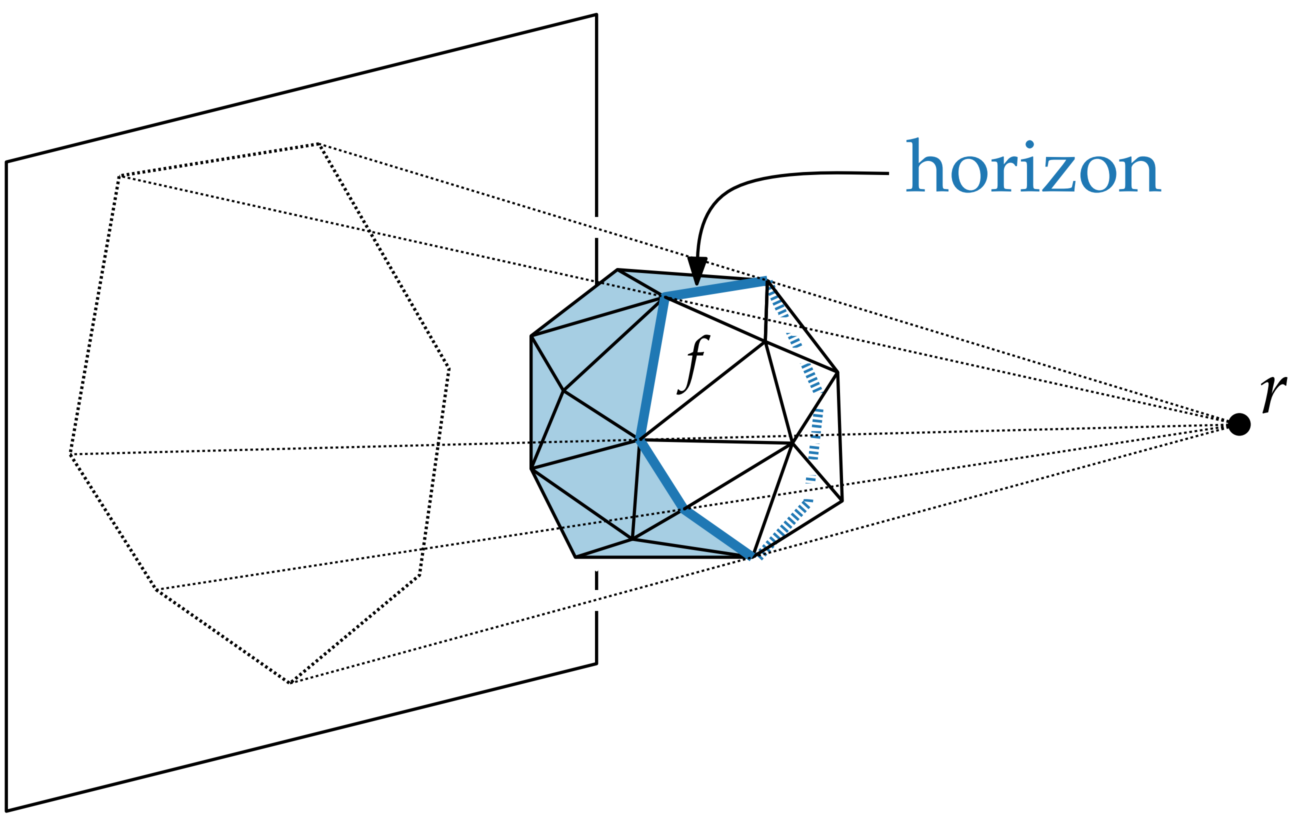


Define conflict graph G:

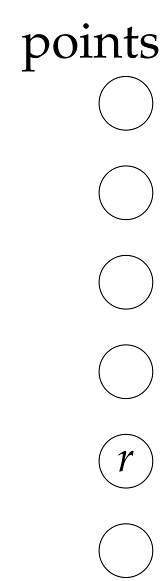
Visibility



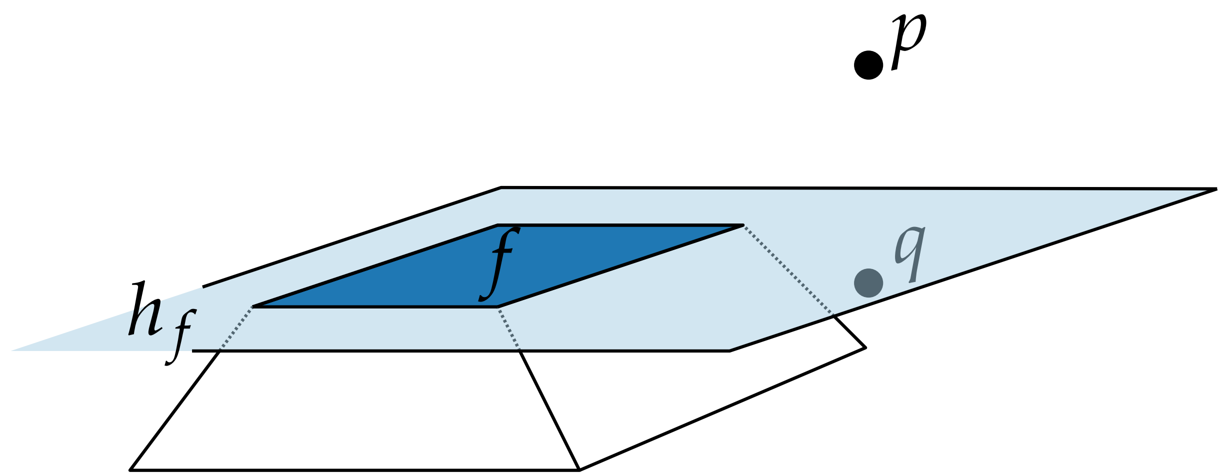
Face f is *visible* from p but not from q.



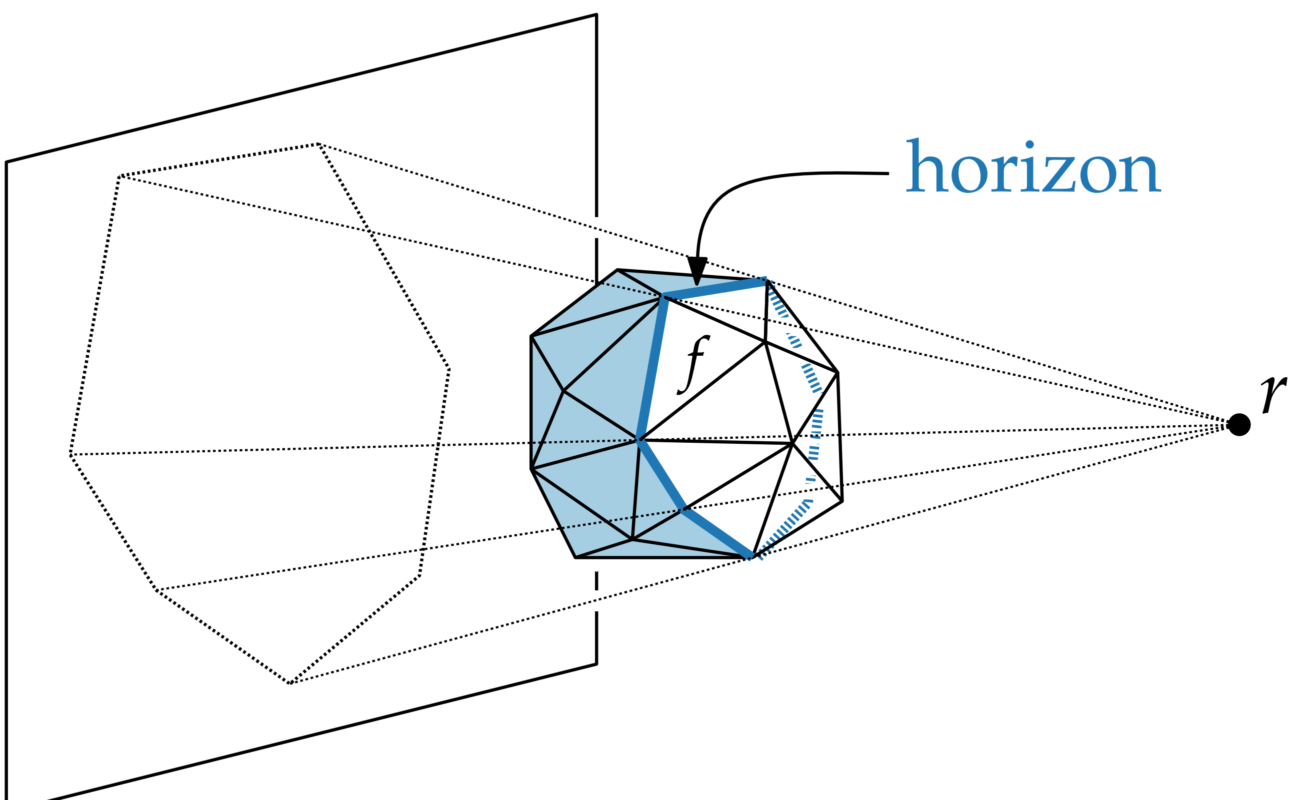
Define conflict graph G:

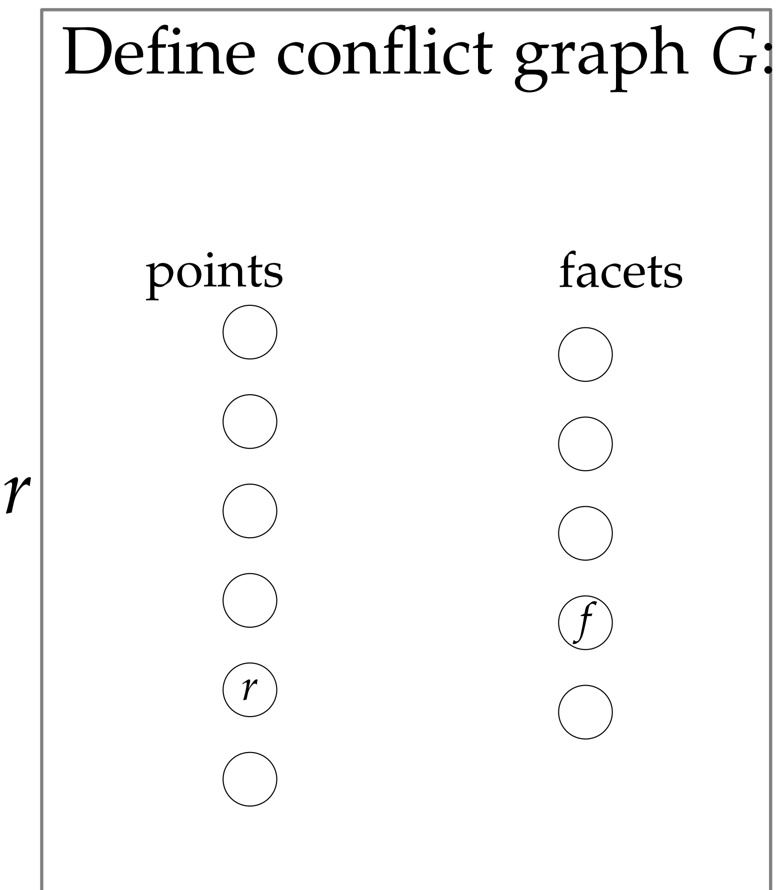


Visibility

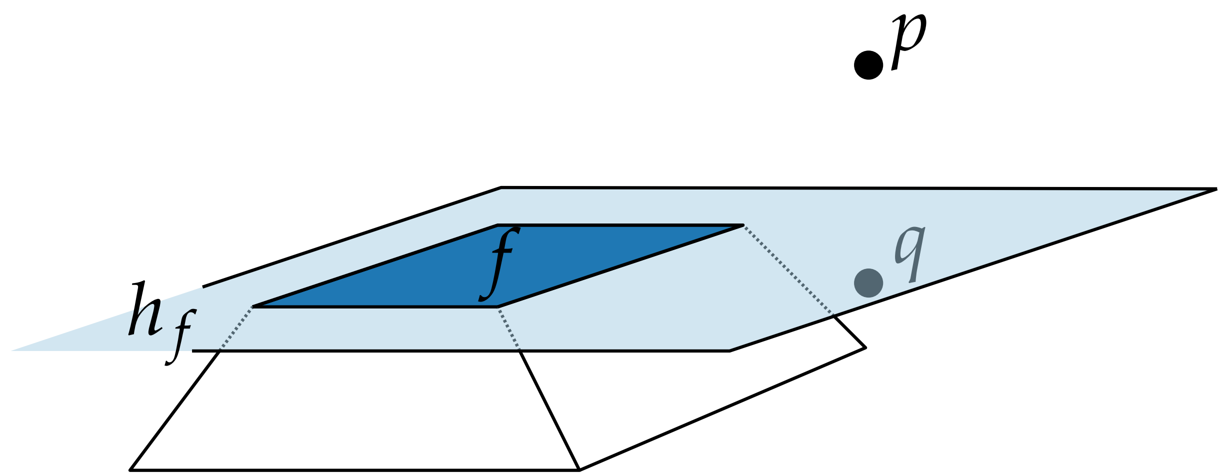


Face f is *visible* from p but not from q.

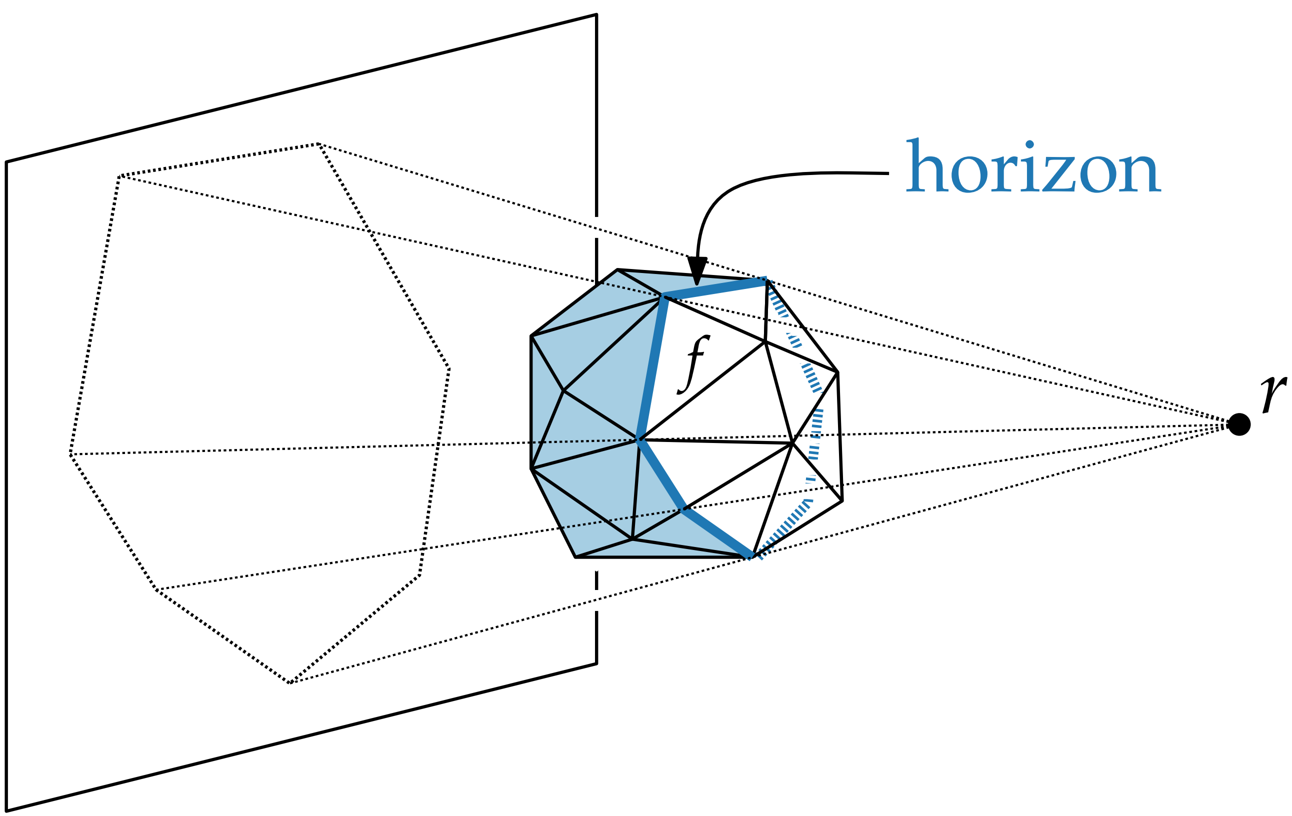




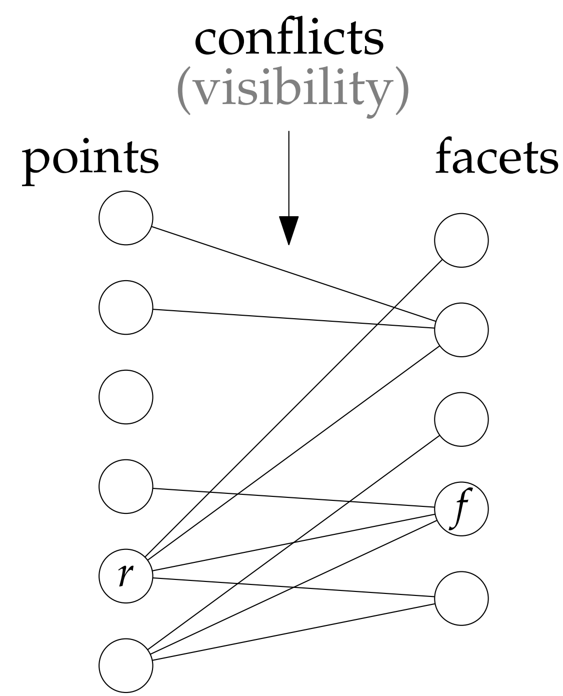
Visibility



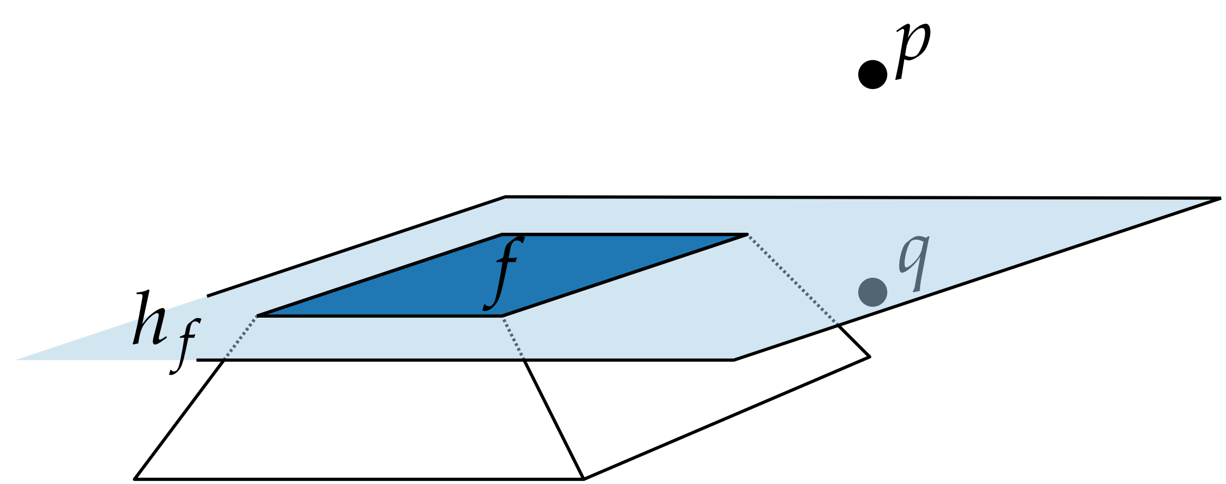
Face f is *visible* from p but not from q.



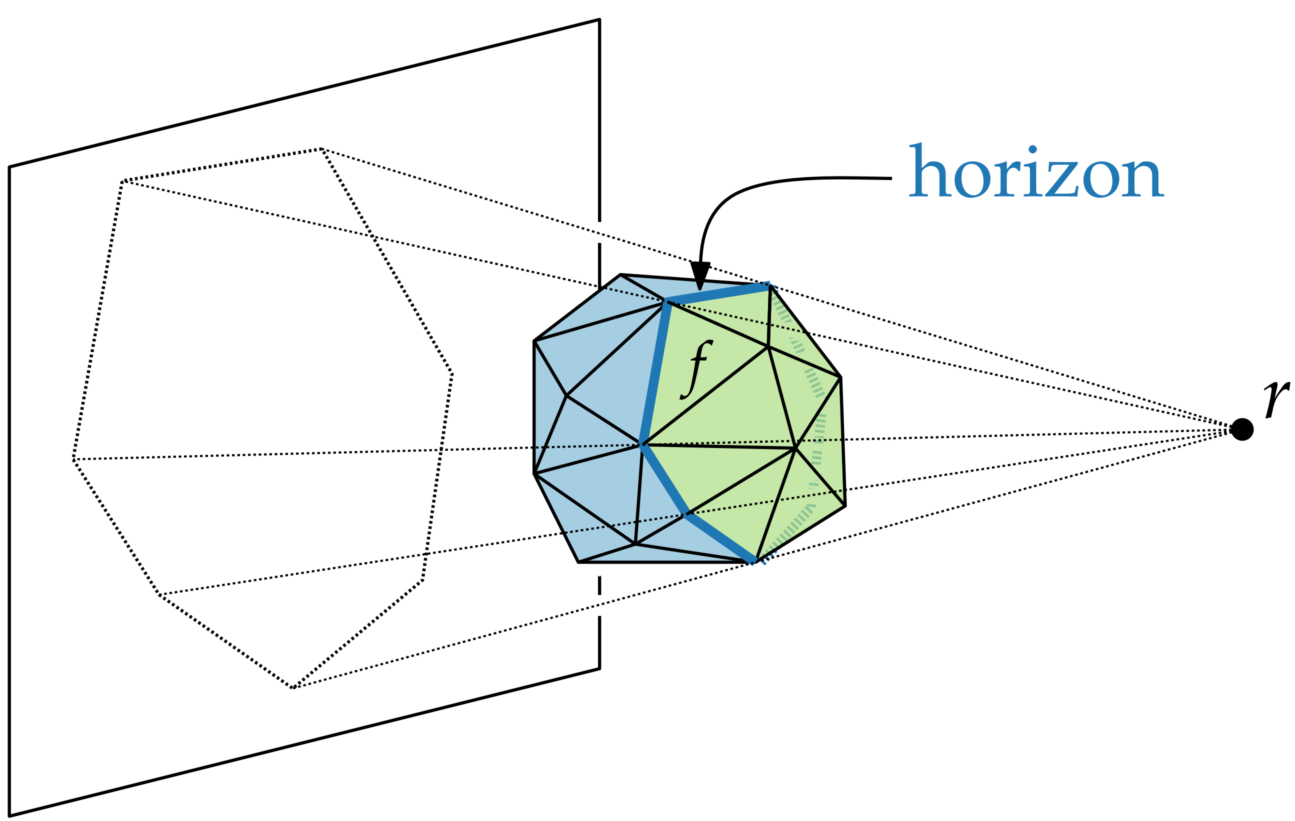
Define conflict graph G:



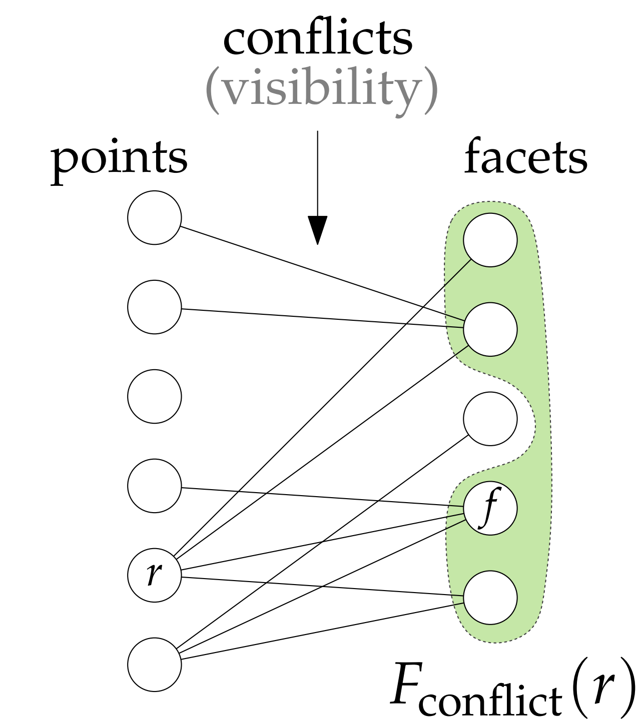
Visibility



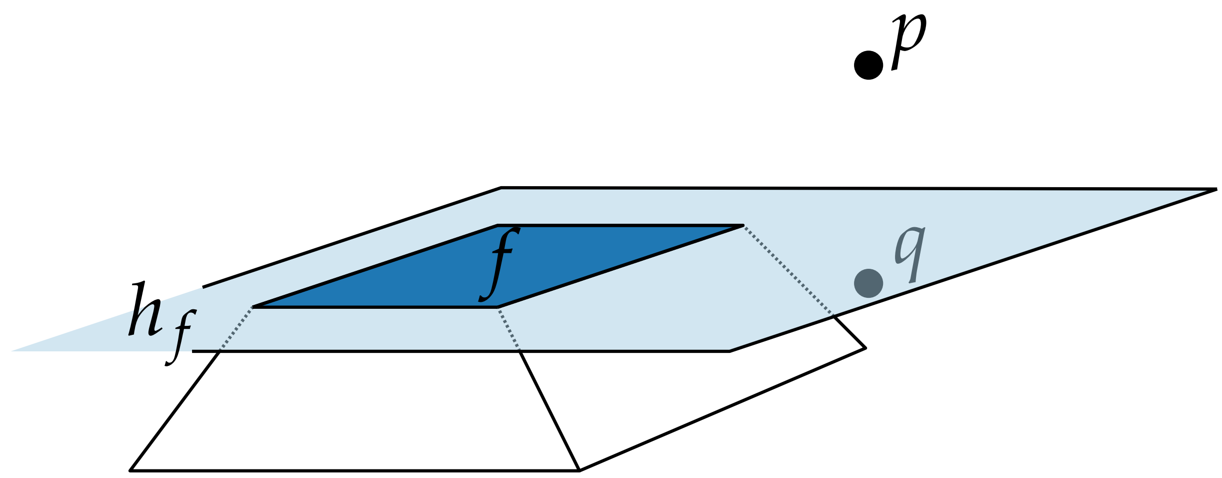
Face f is *visible* from p but not from q.



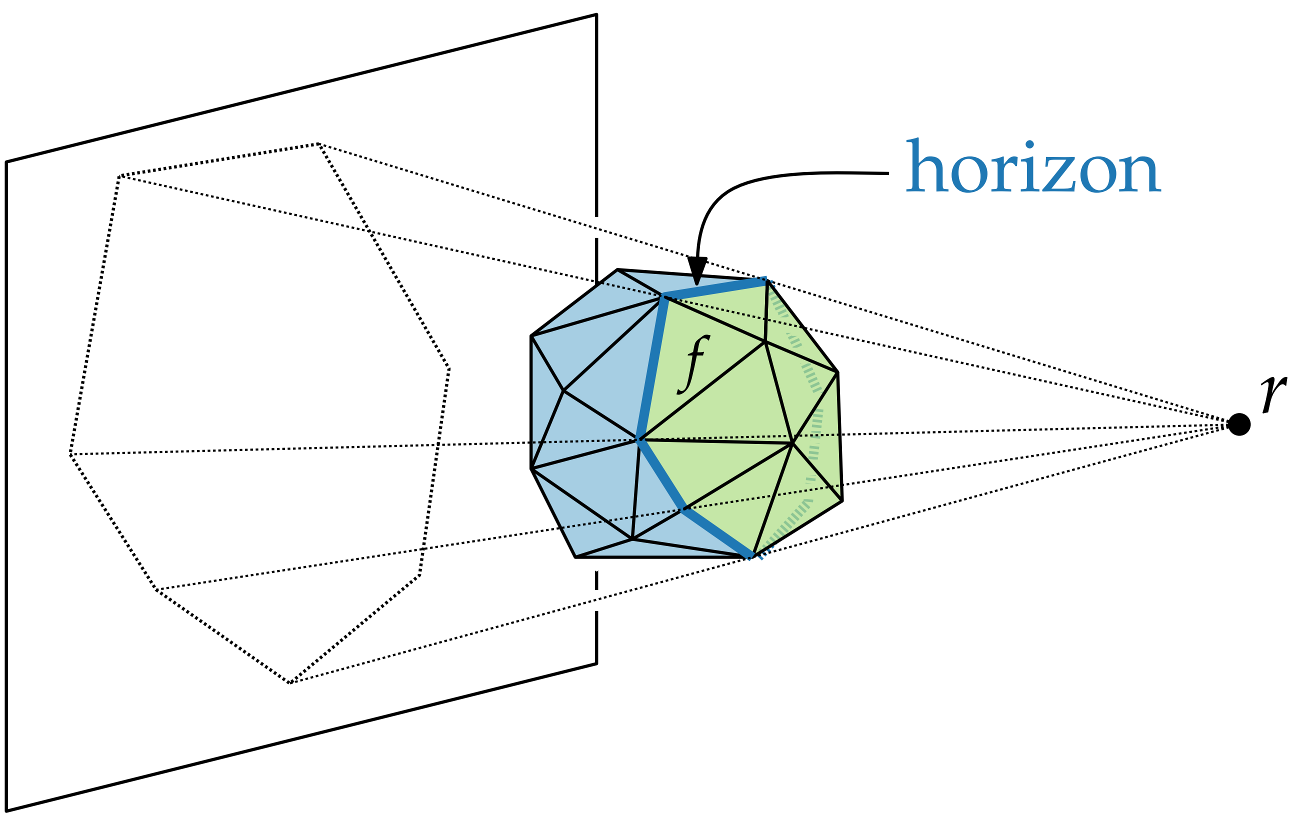
Define conflict graph G:

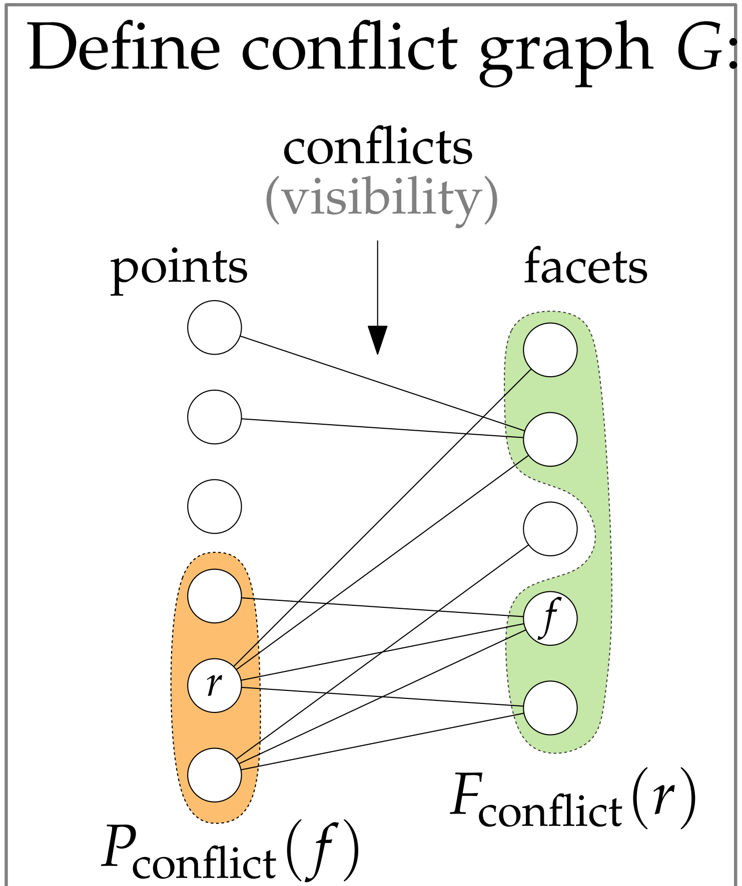


Visibility



Face f is *visible* from p but not from q.





Computational Geometry

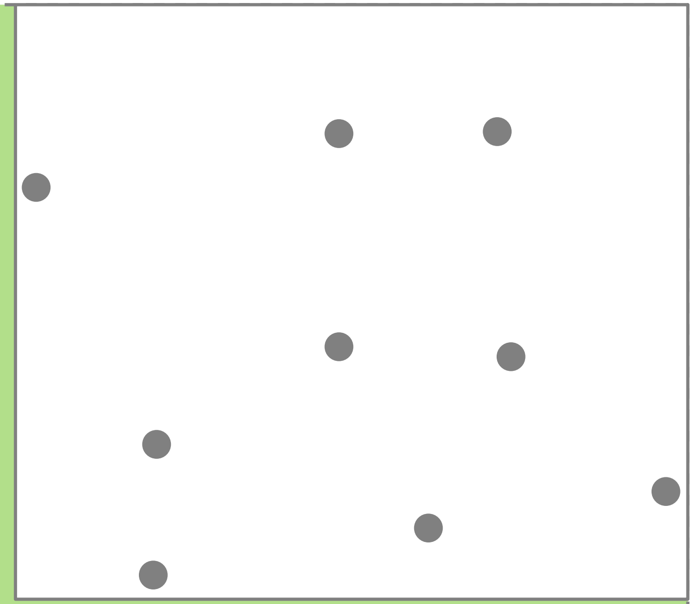
Lecture 9:
Convex Hulls in 3D
or
Mixing More Things

Part II:
Randomized Incremental Algorithm

Philipp Kindermann

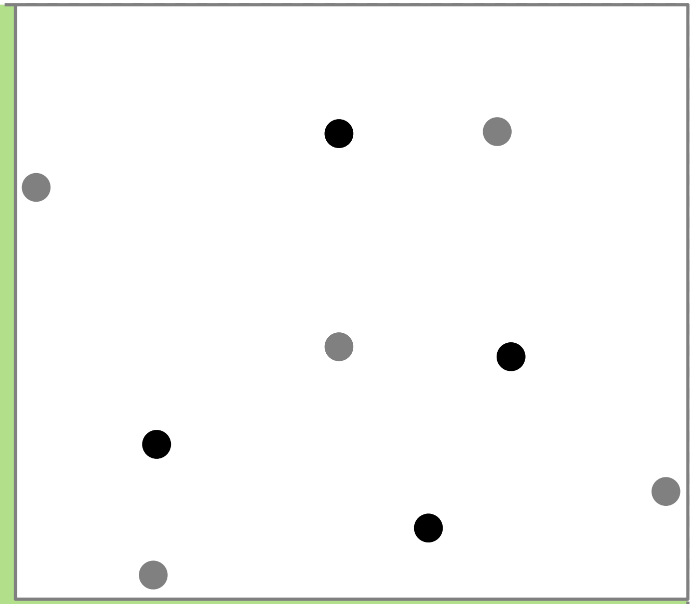
Winter Semester 2020

Rand3DConvexHull($P \subset \mathbb{R}^3$)



Rand3DConvexHull($P \subset \mathbb{R}^3$)

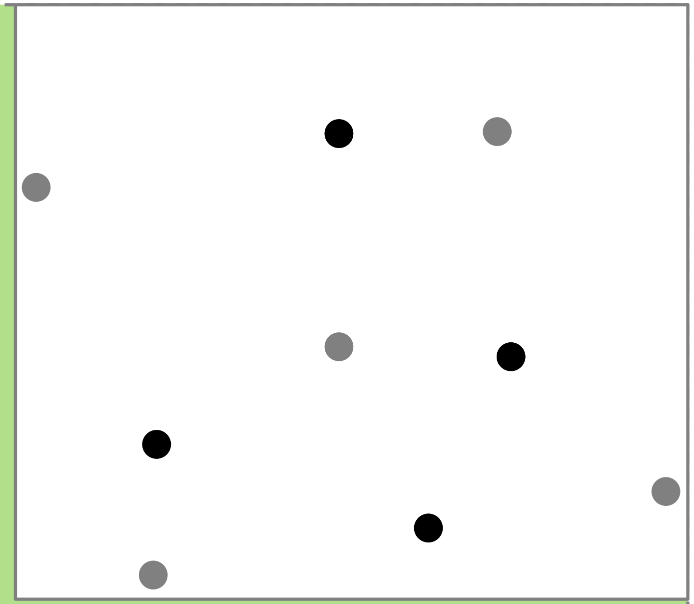
pick non-coplanar set $P' = \{p_1, \ldots, p_4\} \subseteq P$



Rand3DConvexHull($P \subset \mathbb{R}^3$)

pick non-coplanar set $P' = \{p_1, \ldots, p_4\} \subseteq P$

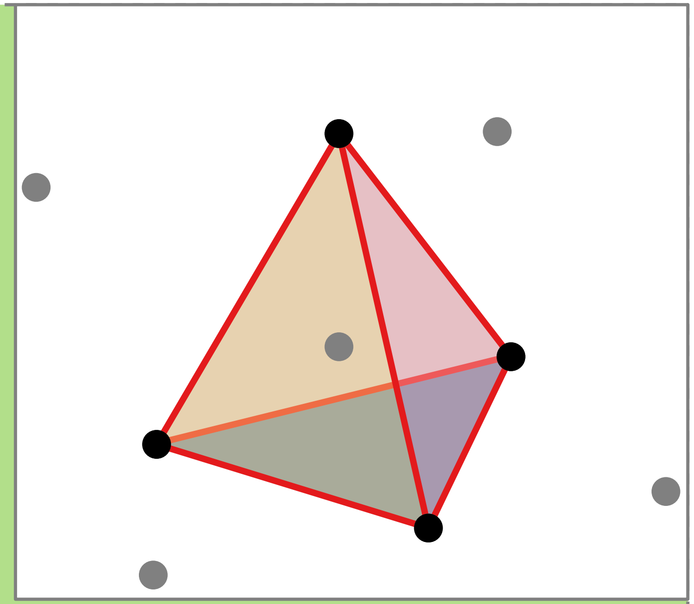
$C \leftarrow \mathrm{CH}(P')$



Rand3DConvexHull($P \subset \mathbb{R}^3$)

pick non-coplanar set $P' = \{p_1, \ldots, p_4\} \subseteq P$

$C \leftarrow \mathrm{CH}(P')$

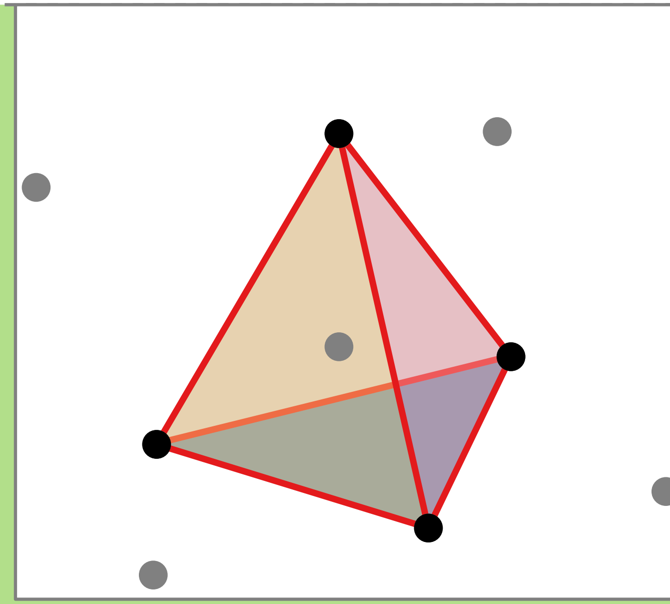


Rand3DConvexHull($P \subset \mathbb{R}^3$)

pick non-coplanar set $P' = \{p_1, \ldots, p_4\} \subseteq P$

$C \leftarrow \mathrm{CH}(P')$

compute rand. perm. $(p_5, \ldots, p_n)$ of $P \setminus P'$

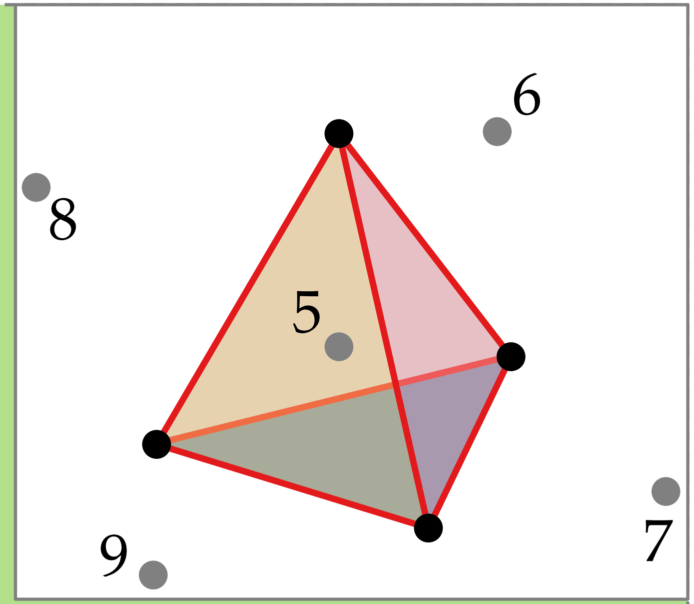


Rand3DConvexHull($P \subset \mathbb{R}^3$)

pick non-coplanar set $P' = \{p_1, \ldots, p_4\} \subseteq P$

$C \leftarrow \mathrm{CH}(P')$

compute rand. perm. $(p_5, \ldots, p_n)$ of $P \setminus P'$



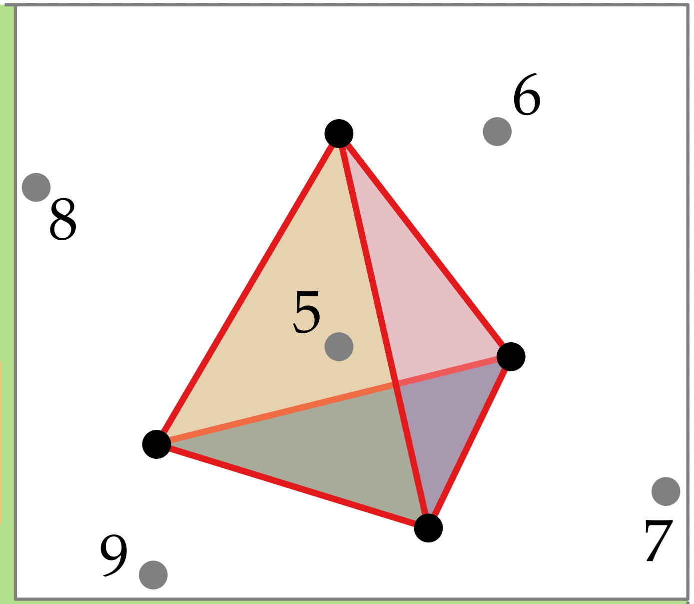
Rand3DConvexHull($P \subset \mathbb{R}^3$)

pick non-coplanar set $P' = \{p_1, \ldots, p_4\} \subseteq P$

$C \leftarrow \mathrm{CH}(P')$

compute rand. perm. $(p_5, \ldots, p_n)$ of $P \setminus P'$

initialize conflict graph G : (p, f) edge $\Leftrightarrow$ f visible from p



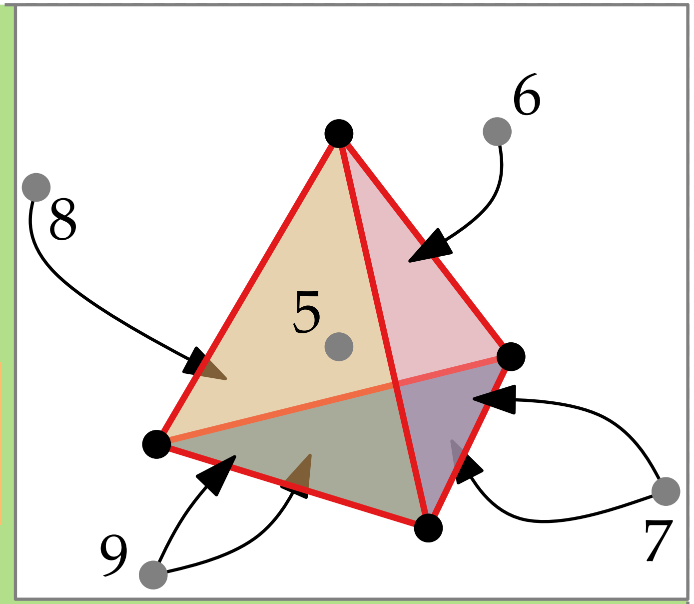
Rand3DConvexHull($P \subset \mathbb{R}^3$)

pick non-coplanar set $P' = \{p_1, \dots, p_4\} \subseteq P$

$C \leftarrow \mathrm{CH}(P')$

compute rand. perm. $(p_5, \dots, p_n)$ of $P \setminus P'$

initialize conflict graph G : (p, f) edge $\Leftrightarrow$ f visible from p



Rand3DConvexHull($P \subset \mathbb{R}^3$)

pick non-coplanar set $P' = \{p_1, \ldots, p_4\} \subseteq P$

$C \leftarrow \mathrm{CH}(P')$

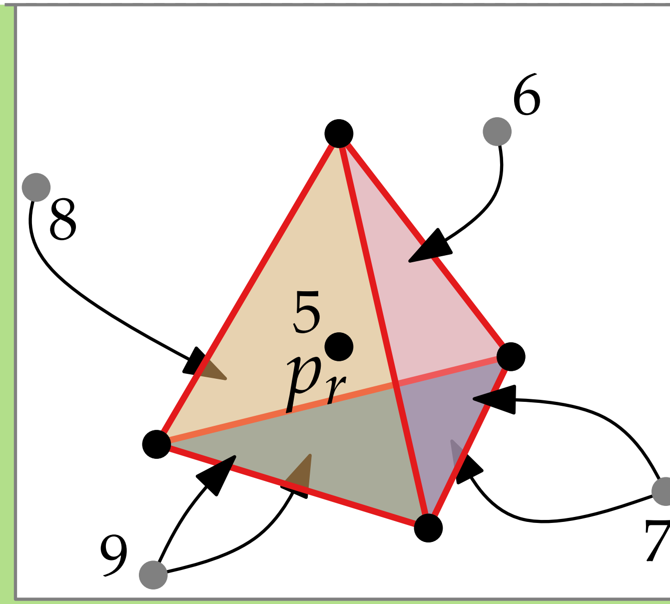
compute rand. perm. $(p_5, \ldots, p_n)$ of $P \setminus P'$

initialize conflict graph G

for $r = 5$ **to** n **do**

if $F_{\text{conflict}}(p_r) \neq \emptyset$ **then** $\{\ p_r \notin C\ \}$

return C



Rand3DConvexHull($P \subset \mathbb{R}^3$)

pick non-coplanar set $P' = \{p_1, \ldots, p_4\} \subseteq P$

$C \leftarrow \mathrm{CH}(P')$

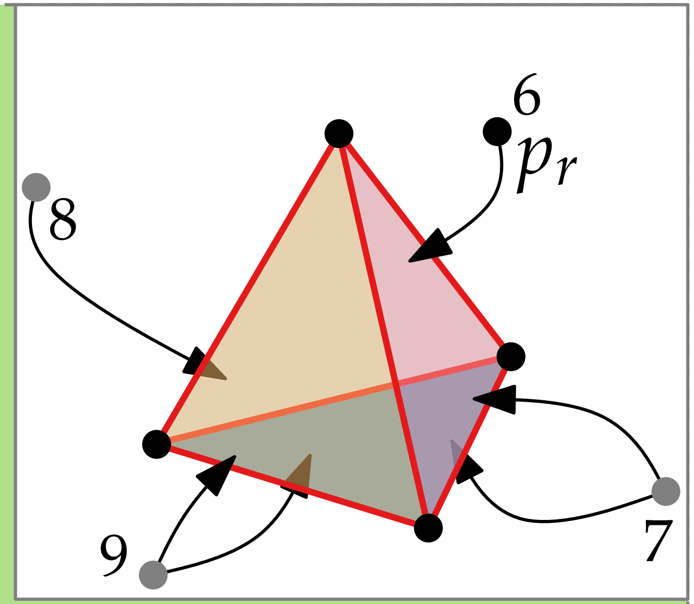
compute rand. perm. $(p_5, \ldots, p_n)$ of $P \setminus P'$

initialize conflict graph G

for $r = 5$ **to** n **do**

  if $F_{\text{conflict}}(p_r) \neq \emptyset$ **then** $\{\ p_r \notin C\ \}$

return C



Rand3DConvexHull($P \subset \mathbb{R}^3$)

pick non-coplanar set $P' = \{p_1, \ldots, p_4\} \subseteq P$

$C \leftarrow \mathrm{CH}(P')$

compute rand. perm. $(p_5, \ldots, p_n)$ of $P \setminus P'$

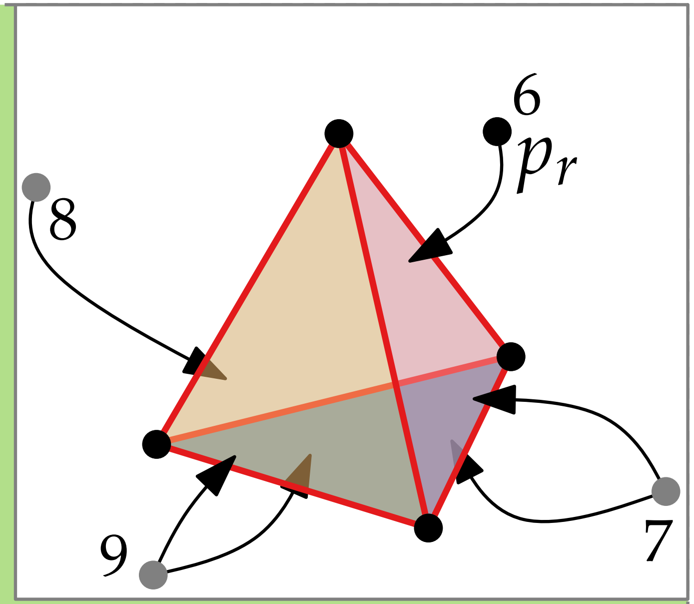
initialize conflict graph G

for $r = 5$ **to** n **do**

  if $F_{\text{conflict}}(p_r) \neq \emptyset$ **then** { $p_r \notin C$ }

    delete all facets in $F_{\text{conflict}}(p_r)$ from C

return C



Rand3DConvexHull($P \subset \mathbb{R}^3$)

pick non-coplanar set $P' = \{p_1, \ldots, p_4\} \subseteq P$

$C \leftarrow \mathrm{CH}(P')$

compute rand. perm. $(p_5, \ldots, p_n)$ of $P \setminus P'$

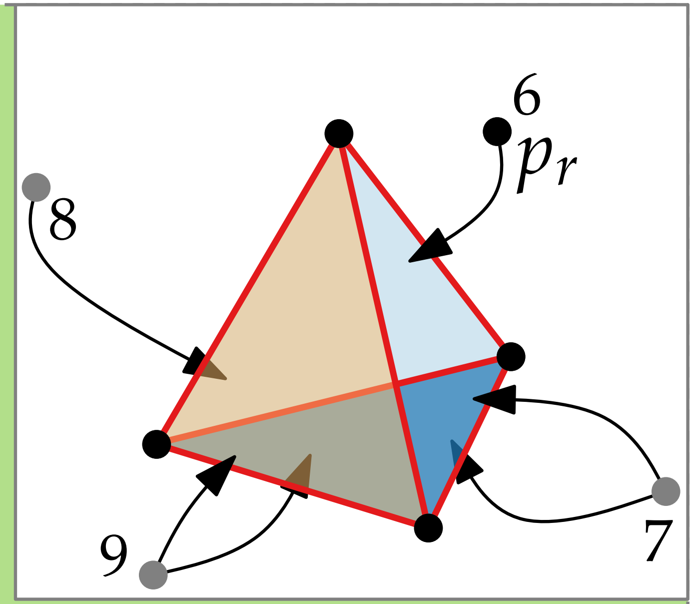
initialize conflict graph G

for $r = 5$ **to** n **do**

  if $F_{\text{conflict}}(p_r) \neq \emptyset$ **then** $\{\ p_r \notin C\ \}$

    delete all facets in $F_{\text{conflict}}(p_r)$ from C

return C



Rand3DConvexHull($P \subset \mathbb{R}^3$)

pick non-coplanar set $P' = \{p_1, \ldots, p_4\} \subseteq P$

$C \leftarrow \mathrm{CH}(P')$

compute rand. perm. $(p_5, \ldots, p_n)$ of $P \setminus P'$

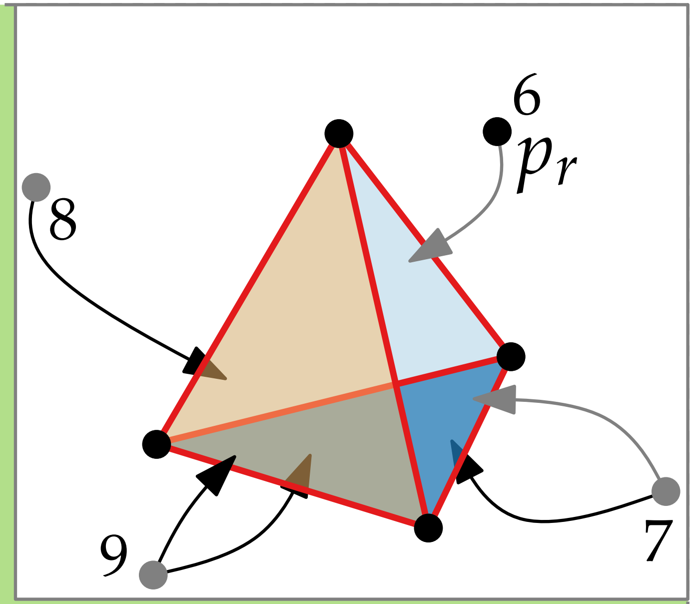
initialize conflict graph G

for $r = 5$ **to** n **do**

    if $F_{\text{conflict}}(p_r) \neq \emptyset$ **then** $\{ p_r \notin C \}$

        delete all facets in $F_{\text{conflict}}(p_r)$ from C

return C



Rand3DConvexHull($P \subset \mathbb{R}^3$)

pick non-coplanar set $P' = \{p_1, \ldots, p_4\} \subseteq P$

$C \leftarrow \mathrm{CH}(P')$

compute rand. perm. $(p_5, \ldots, p_n)$ of $P \setminus P'$

initialize conflict graph G

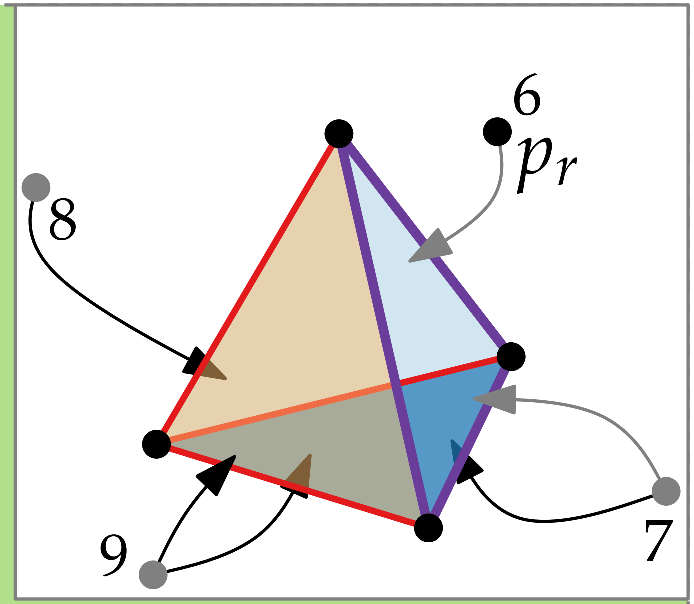
for $r = 5$ **to** n **do**

  if $F_{\text{conflict}}(p_r) \neq \emptyset$ **then** $\{\ p_r \notin C\ \}$

    delete all facets in $F_{\text{conflict}}(p_r)$ from C

    $\mathcal{L} \leftarrow$ list of horizon edges visible from p_r

return C



Rand3DConvexHull($P \subset \mathbb{R}^3$)

pick non-coplanar set $P' = \{p_1, \ldots, p_4\} \subseteq P$

$C \leftarrow \mathrm{CH}(P')$

compute rand. perm. $(p_5, \ldots, p_n)$ of $P \setminus P'$

initialize conflict graph G

for $r = 5$ **to** n **do**

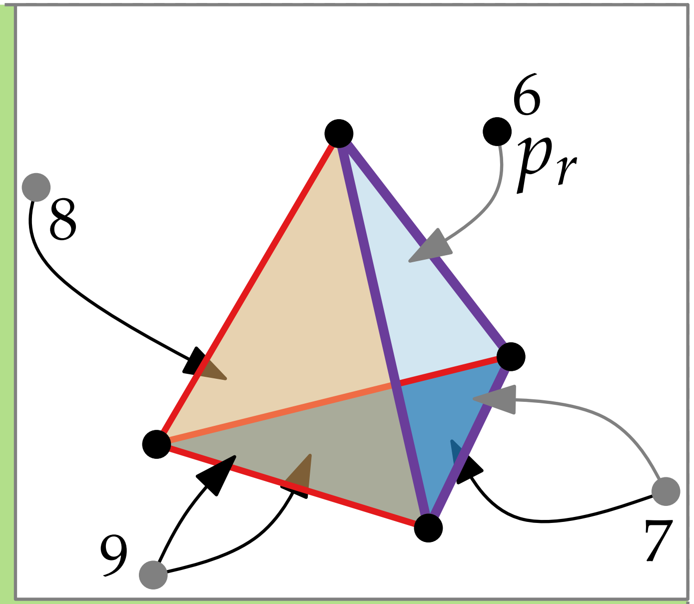
  if $F_{\mathrm{conflict}}(p_r) \neq \emptyset$ **then** $\{\ p_r \notin C\ \}$

    delete all facets in $F_{\mathrm{conflict}}(p_r)$ from C

    $\mathcal{L} \leftarrow$ list of horizon edges visible from p_r

    foreach $e \in \mathcal{L}$ **do**

return C



Rand3DConvexHull($P \subset \mathbb{R}^3$)

pick non-coplanar set $P' = \{p_1, \ldots, p_4\} \subseteq P$

$C \leftarrow \mathrm{CH}(P')$

compute rand. perm. $(p_5, \ldots, p_n)$ of $P \setminus P'$

initialize conflict graph G

for $r = 5$ **to** n **do**

  if $F_{\text{conflict}}(p_r) \neq \emptyset$ **then** $\{\ p_r \notin C\ \}$

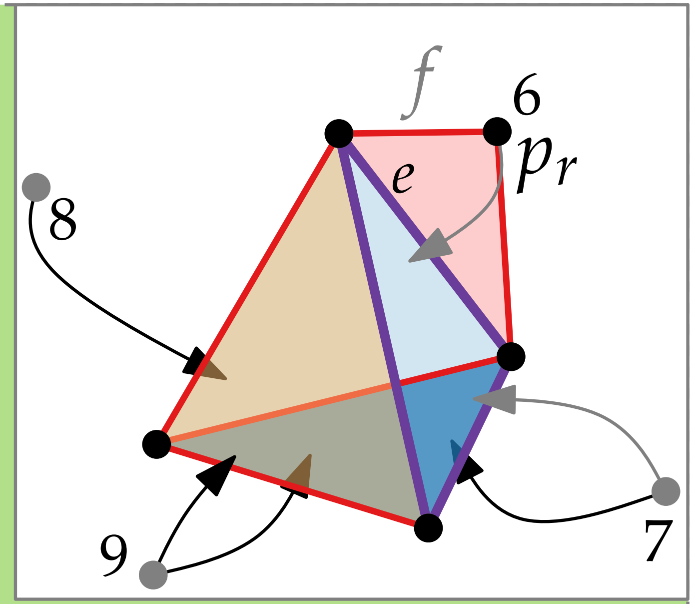
    delete all facets in $F_{\text{conflict}}(p_r)$ from C

    $\mathcal{L} \leftarrow$ list of horizon edges visible from p_r

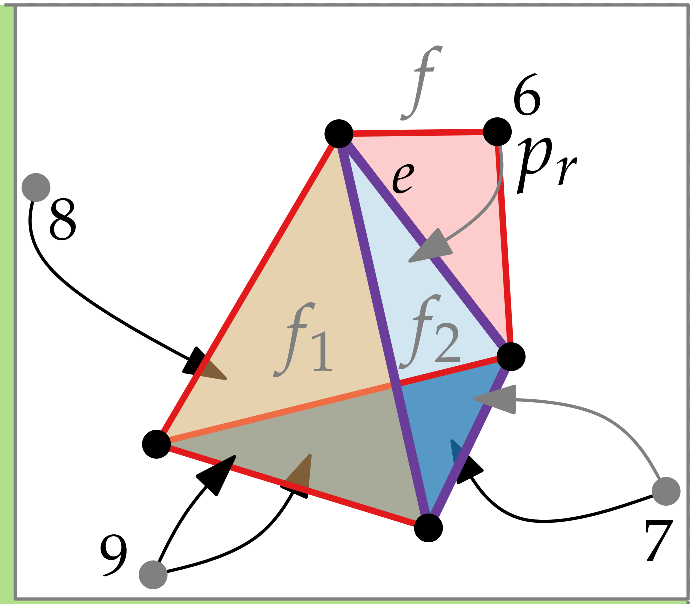
    foreach $e \in \mathcal{L}$ **do**

      $f \leftarrow C$.create_facet(e, p_r); create vtx for f in G

return C



Rand3DConvexHull$(P \subset \mathbb{R}^3)$




```
pick non-coplanar set P' = {p_1, ..., p_4} ⊆ P
C ← CH(P')
compute rand. perm. (p_5, ..., p_n) of P\P'
initialize conflict graph G
for r = 5 to n do
    if F_conflict(p_r) ≠ ∅ then  { p_r ∉ C }
        delete all facets in F_conflict(p_r) from C
        ℒ ← list of horizon edges visible from p_r
        foreach e ∈ ℒ do
            f ← C.create_facet(e, p_r); create vtx for f in G
            (f_1, f_2) ← previously_incident_C(e)
return C
```

Rand3DConvexHull($P \subset \mathbb{R}^3$)

pick non-coplanar set $P' = \{p_1, \ldots, p_4\} \subseteq P$

$C \leftarrow \mathrm{CH}(P')$

compute rand. perm. $(p_5, \ldots, p_n)$ of $P \setminus P'$

initialize conflict graph G

for $r = 5$ **to** n **do**

  if $F_{\text{conflict}}(p_r) \neq \emptyset$ **then** $\{\ p_r \notin C\ \}$

    delete all facets in $F_{\text{conflict}}(p_r)$ from C

    $\mathcal{L} \leftarrow$ list of horizon edges visible from p_r

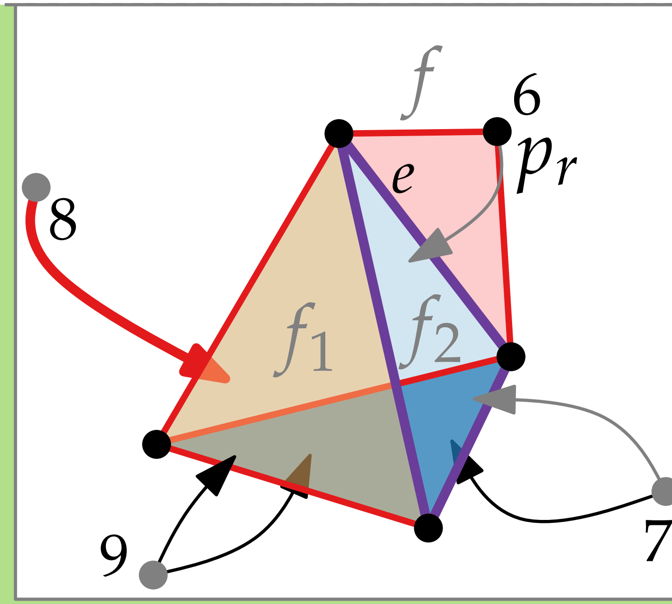
    foreach $e \in \mathcal{L}$ **do**

      $f \leftarrow C$.create_facet(e, p_r); create vtx for f in G

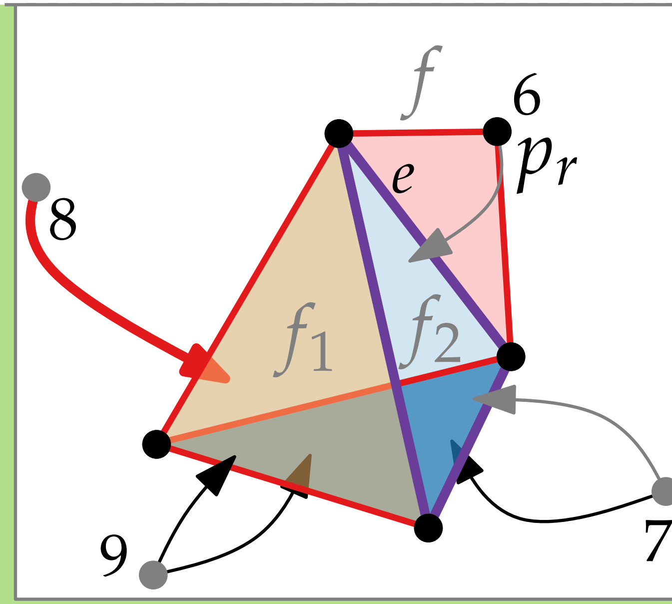
      $(f_1, f_2) \leftarrow$ previously_incident$_C(e)$

      $P(e) \leftarrow P_{\text{conflict}}(f_1) \cup P_{\text{conflict}}(f_2)$

return C



Rand3DConvexHull($P \subset \mathbb{R}^3$)



pick non-coplanar set $P' = \{p_1, \ldots, p_4\} \subseteq P$

$C \leftarrow \mathrm{CH}(P')$

compute rand. perm. $(p_5, \ldots, p_n)$ of $P \setminus P'$

initialize conflict graph G

for $r = 5$ **to** n **do**

  if $F_{\text{conflict}}(p_r) \neq \emptyset$ **then** $\{ \, p_r \notin C \, \}$

    delete all facets in $F_{\text{conflict}}(p_r)$ from C

    $\mathcal{L} \leftarrow$ list of horizon edges visible from p_r

    foreach $e \in \mathcal{L}$ **do**

      $f \leftarrow C$.create_facet(e, p_r); create vtx for f in G

      $(f_1, f_2) \leftarrow$ previously_incident$_C(e)$

      $P(e) \leftarrow P_{\text{conflict}}(f_1) \cup P_{\text{conflict}}(f_2)$

      foreach $p \in P(e)$ **do**

return C

Rand3DConvexHull($P \subset \mathbb{R}^3$)

pick non-coplanar set $P' = \{p_1, \dots, p_4\} \subseteq P$

$C \leftarrow \mathrm{CH}(P')$

compute rand. perm. $(p_5, \dots, p_n)$ of $P \setminus P'$

initialize conflict graph G

for $r = 5$ **to** n **do**

  if $F_{\text{conflict}}(p_r) \neq \emptyset$ **then** $\{ p_r \notin C \}$

    delete all facets in $F_{\text{conflict}}(p_r)$ from C

    $\mathcal{L} \leftarrow$ list of horizon edges visible from p_r

    foreach $e \in \mathcal{L}$ **do**

      $f \leftarrow C$.create_facet(e, p_r); create vtx for f in G

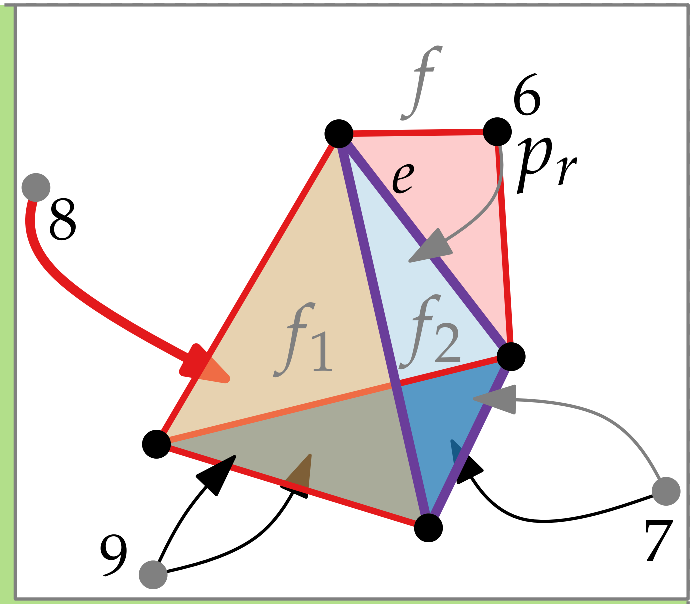
      $(f_1, f_2) \leftarrow$ previously_incident$_C(e)$

      $P(e) \leftarrow P_{\text{conflict}}(f_1) \cup P_{\text{conflict}}(f_2)$

      foreach $p \in P(e)$ **do**

        if f is visible from p

return C



Rand3DConvexHull($P \subset \mathbb{R}^3$)

pick non-coplanar set $P' = \{p_1, \ldots, p_4\} \subseteq P$

$C \leftarrow \mathrm{CH}(P')$

compute rand. perm. $(p_5, \ldots, p_n)$ of $P \setminus P'$

initialize conflict graph G

for $r = 5$ **to** n **do**

  if $F_{\text{conflict}}(p_r) \neq \emptyset$ **then** $\{ p_r \notin C \}$

    delete all facets in $F_{\text{conflict}}(p_r)$ from C

    $\mathcal{L} \leftarrow$ list of horizon edges visible from p_r

    foreach $e \in \mathcal{L}$ **do**

      $f \leftarrow C$.create_facet(e, p_r); create vtx for f in G

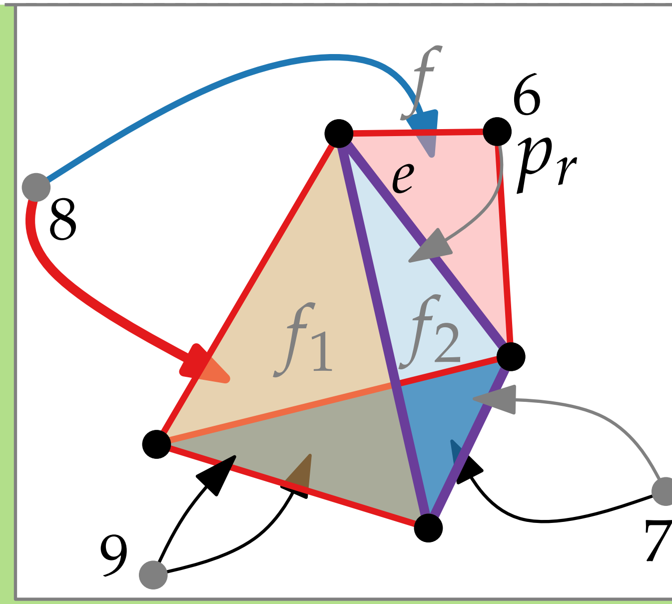
      $(f_1, f_2) \leftarrow$ previously_incident$_C(e)$

      $P(e) \leftarrow P_{\text{conflict}}(f_1) \cup P_{\text{conflict}}(f_2)$

      foreach $p \in P(e)$ **do**

        if f is visible from p **then** add edge (p, f) to G

return C



Rand3DConvexHull($P \subset \mathbb{R}^3$)

pick non-coplanar set $P' = \{p_1, \ldots, p_4\} \subseteq P$

$C \leftarrow \mathrm{CH}(P')$

compute rand. perm. $(p_5, \ldots, p_n)$ of $P \setminus P'$

initialize conflict graph G

for $r = 5$ **to** n **do**

  if $F_{\text{conflict}}(p_r) \neq \emptyset$ **then** $\{ p_r \notin C \}$

    delete all facets in $F_{\text{conflict}}(p_r)$ from C

    $\mathcal{L} \leftarrow$ list of horizon edges visible from p_r

    foreach $e \in \mathcal{L}$ **do**

      $f \leftarrow C$.create_facet(e, p_r); create vtx for f in G

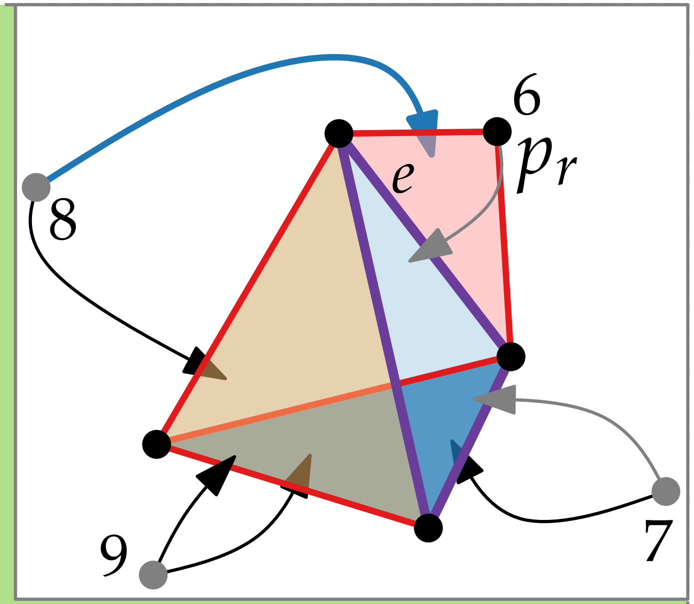
      $(f_1, f_2) \leftarrow$ previously_incident$_C(e)$

      $P(e) \leftarrow P_{\text{conflict}}(f_1) \cup P_{\text{conflict}}(f_2)$

      foreach $p \in P(e)$ **do**

        if f is visible from p **then** add edge (p, f) to G

return C



Rand3DConvexHull($P \subset \mathbb{R}^3$)

pick non-coplanar set $P' = \{p_1, \ldots, p_4\} \subseteq P$

$C \leftarrow \mathrm{CH}(P')$

compute rand. perm. $(p_5, \ldots, p_n)$ of $P \setminus P'$

initialize conflict graph G

for $r = 5$ **to** n **do**

  if $F_{\mathrm{conflict}}(p_r) \neq \emptyset$ **then** $\{ p_r \notin C \}$

    delete all facets in $F_{\mathrm{conflict}}(p_r)$ from C

    $\mathcal{L} \leftarrow$ list of horizon edges visible from p_r

    foreach $e \in \mathcal{L}$ **do**

      $f \leftarrow C$.create_facet(e, p_r); create vtx for f in G

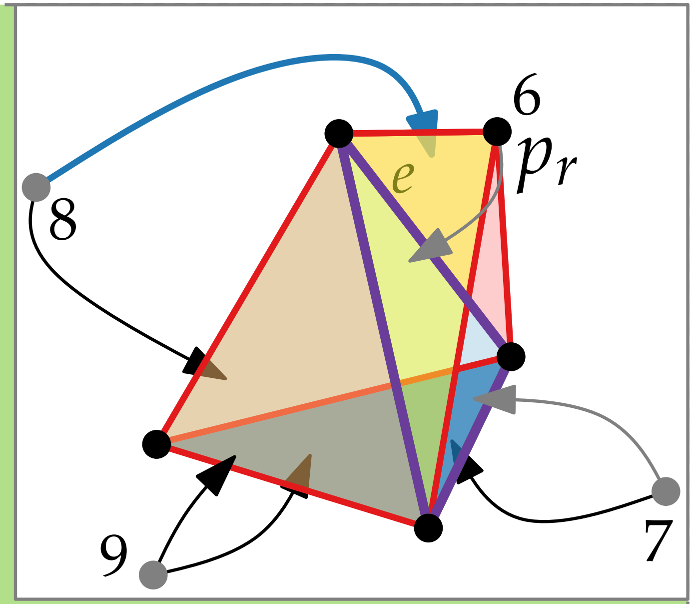
      $(f_1, f_2) \leftarrow$ previously_incident$_C(e)$

      $P(e) \leftarrow P_{\mathrm{conflict}}(f_1) \cup P_{\mathrm{conflict}}(f_2)$

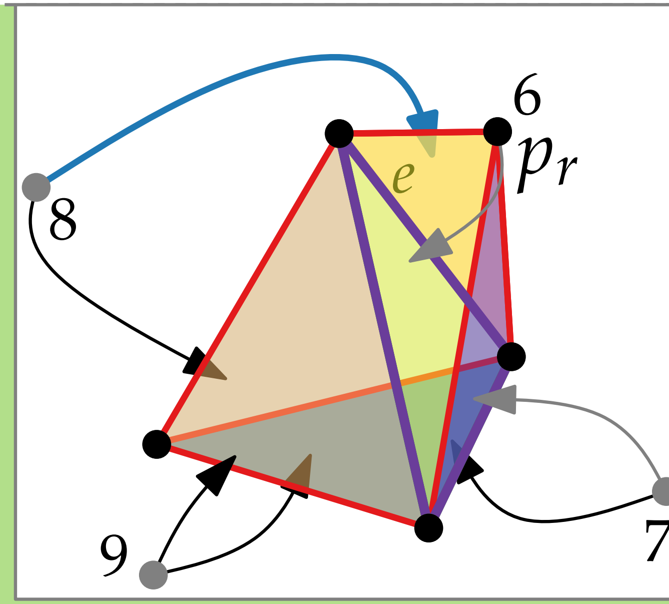
      foreach $p \in P(e)$ **do**

        if f is visible from p **then** add edge (p, f) to G

return C



Rand3DConvexHull($P \subset \mathbb{R}^3$)



pick non-coplanar set $P' = \{p_1, \ldots, p_4\} \subseteq P$

$C \leftarrow \mathrm{CH}(P')$

compute rand. perm. $(p_5, \ldots, p_n)$ of $P \setminus P'$

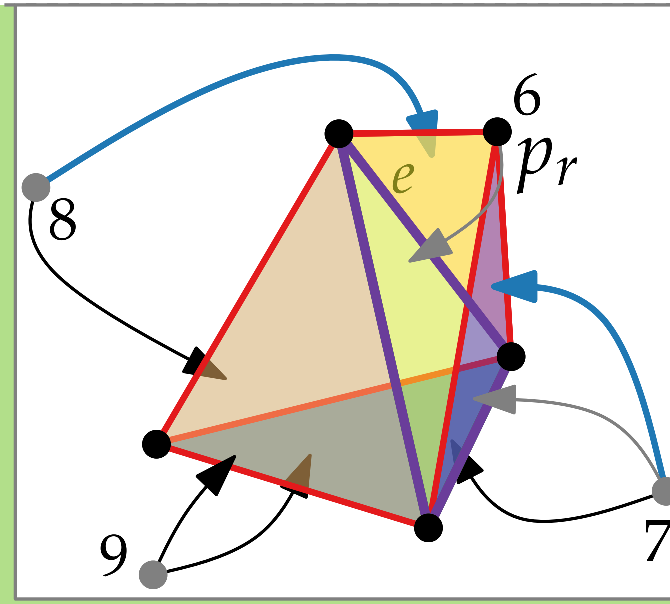
initialize conflict graph G

for $r = 5$ **to** n **do**

- **if** $F_{\mathrm{conflict}}(p_r) \neq \emptyset$ **then** $\{\ p_r \notin C\ \}$
  - delete all facets in $F_{\mathrm{conflict}}(p_r)$ from C
  - $\mathcal{L} \leftarrow$ list of horizon edges visible from p_r
  - **foreach** $e \in \mathcal{L}$ **do**
    - $f \leftarrow C$.create_facet(e, p_r); create vtx for f in G
    - $(f_1, f_2) \leftarrow$ previously_incident$_C(e)$
    - $P(e) \leftarrow P_{\mathrm{conflict}}(f_1) \cup P_{\mathrm{conflict}}(f_2)$
    - **foreach** $p \in P(e)$ **do**
      - **if** f is visible from p **then** add edge (p, f) to G

return C

Rand3DConvexHull($P \subset \mathbb{R}^3$)



pick non-coplanar set $P' = \{p_1, \ldots, p_4\} \subseteq P$

$C \leftarrow \mathrm{CH}(P')$

compute rand. perm. $(p_5, \ldots, p_n)$ of $P \backslash P'$

initialize conflict graph G

for $r = 5$ **to** n **do**

  if $F_{\text{conflict}}(p_r) \neq \emptyset$ **then** $\{ p_r \notin C \}$

    delete all facets in $F_{\text{conflict}}(p_r)$ from C

    $\mathcal{L} \leftarrow$ list of horizon edges visible from p_r

    foreach $e \in \mathcal{L}$ **do**

      $f \leftarrow C$.create_facet(e, p_r); create vtx for f in G

      $(f_1, f_2) \leftarrow$ previously_incident$_C(e)$

      $P(e) \leftarrow P_{\text{conflict}}(f_1) \cup P_{\text{conflict}}(f_2)$

      foreach $p \in P(e)$ **do**

        if f is visible from p **then** add edge (p, f) to G

return C

Rand3DConvexHull($P \subset \mathbb{R}^3$)

pick non-coplanar set $P' = \{p_1, \ldots, p_4\} \subseteq P$

$C \leftarrow \mathrm{CH}(P')$

compute rand. perm. $(p_5, \ldots, p_n)$ of $P \setminus P'$

initialize conflict graph G

for $r = 5$ **to** n **do**

  if $F_{\text{conflict}}(p_r) \neq \emptyset$ **then** $\{\ p_r \notin C\ \}$

    delete all facets in $F_{\text{conflict}}(p_r)$ from C

    $\mathcal{L} \leftarrow$ list of horizon edges visible from p_r

    foreach $e \in \mathcal{L}$ **do**

      $f \leftarrow C$.create_facet(e, p_r); create vtx for f in G

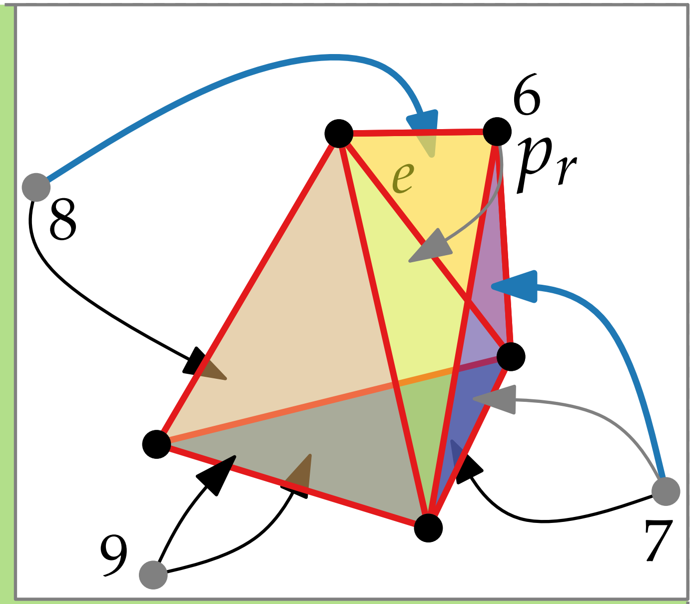
      $(f_1, f_2) \leftarrow$ previously_incident$_C(e)$

      $P(e) \leftarrow P_{\text{conflict}}(f_1) \cup P_{\text{conflict}}(f_2)$

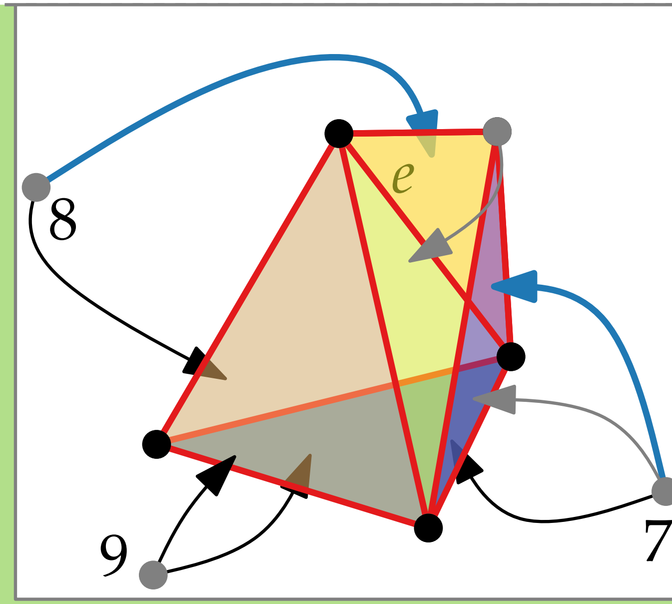
      foreach $p \in P(e)$ **do**

        if f is visible from p **then** add edge (p, f) to G

return C

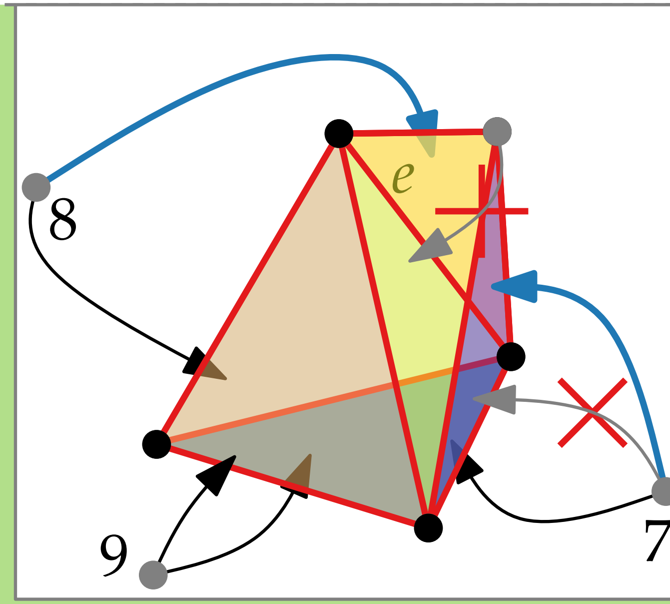


```
Rand3DConvexHull(P ⊂ ℝ³)
  pick non-coplanar set P′ = {p_1, …, p_4} ⊆ P
  C ← CH(P′)
  compute rand. perm. (p_5, …, p_n) of P\P′
  initialize conflict graph G
  for r = 5 to n do
      if F_conflict(p_r) ≠ ∅ then  { p_r ∉ C }
          delete all facets in F_conflict(p_r) from C
          ℒ ← list of horizon edges visible from p_r
          foreach e ∈ ℒ do
              f ← C.create_facet(e, p_r); create vtx for f in G
              (f_1, f_2) ← previously_incident_C(e)
              P(e) ← P_conflict(f_1) ∪ P_conflict(f_2)
              foreach p ∈ P(e) do
                  if f is visible from p then add edge (p, f) to G
          delete vtc {p_r} ∪ F_conflict(p_r) from G
  return C
```



Rand3DConvexHull($P \subset \mathbb{R}^3$)

```
pick non-coplanar set P′ = {p_1, …, p_4} ⊆ P
C ← CH(P′)
compute rand. perm. (p_5, …, p_n) of P \ P′
initialize conflict graph G
for r = 5 to n do
    if F_conflict(p_r) ≠ ∅ then  { p_r ∉ C }
        delete all facets in F_conflict(p_r) from C
        ℒ ← list of horizon edges visible from p_r
        foreach e ∈ ℒ do
            f ← C.create_facet(e, p_r); create vtx for f in G
            (f_1, f_2) ← previously_incident_C(e)
            P(e) ← P_conflict(f_1) ∪ P_conflict(f_2)
            foreach p ∈ P(e) do
                if f is visible from p then add edge (p, f) to G
        delete vtc {p_r} ∪ F_conflict(p_r) from G
return C
```



Rand3DConvexHull($P \subset \mathbb{R}^3$)

pick non-coplanar set $P' = \{p_1, \ldots, p_4\} \subseteq P$

$C \leftarrow \mathrm{CH}(P')$

compute rand. perm. $(p_5, \ldots, p_n)$ of $P \setminus P'$

initialize conflict graph G

for $r = 5$ **to** n **do**

  if $F_{\text{conflict}}(p_r) \neq \emptyset$ **then** $\{\ p_r \notin C\ \}$

    delete all facets in $F_{\text{conflict}}(p_r)$ from C

    $\mathcal{L} \leftarrow$ list of horizon edges visible from p_r

    foreach $e \in \mathcal{L}$ **do**

      $f \leftarrow C.\text{create\_facet}(e, p_r)$; create vtx for f in G

      $(f_1, f_2) \leftarrow \text{previously\_incident}_C(e)$

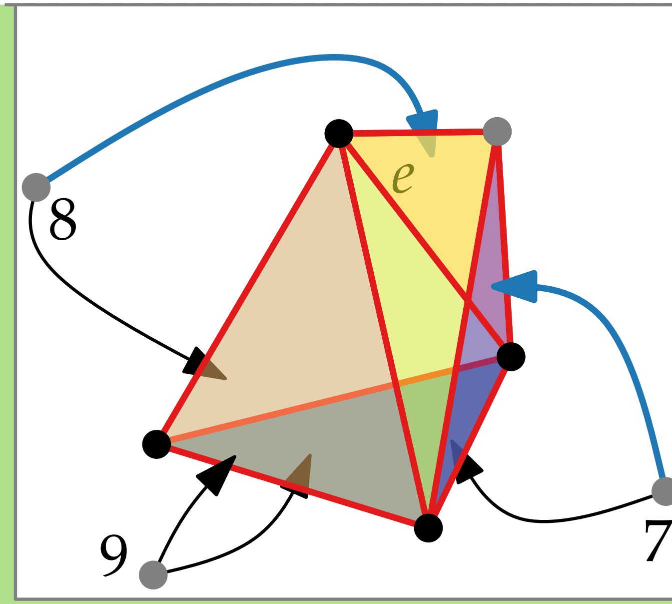
      $P(e) \leftarrow P_{\text{conflict}}(f_1) \cup P_{\text{conflict}}(f_2)$

      foreach $p \in P(e)$ **do**

        if f is visible from p **then** add edge (p, f) to G

    delete vtc $\{p_r\} \cup F_{\text{conflict}}(p_r)$ from G

return C



Rand3DConvexHull($P \subset \mathbb{R}^3$)

pick non-coplanar set $P' = \{p_1, \ldots, p_4\} \subseteq P$

$C \leftarrow \mathrm{CH}(P')$

compute rand. perm. $(p_5, \ldots, p_n)$ of $P \setminus P'$

initialize conflict graph G

for $r = 5$ **to** n **do**

    if $F_{\mathrm{conflict}}(p_r) \neq \emptyset$ **then** $\{ p_r \notin C \}$

        delete all facets in $F_{\mathrm{conflict}}(p_r)$ from C

        $\mathcal{L} \leftarrow$ list of horizon edges visible from p_r

        foreach $e \in \mathcal{L}$ **do**

            $f \leftarrow C.\mathrm{create\_facet}(e, p_r)$; create vtx for f in G

            $(f_1, f_2) \leftarrow \mathrm{previously\_incident}_C(e)$

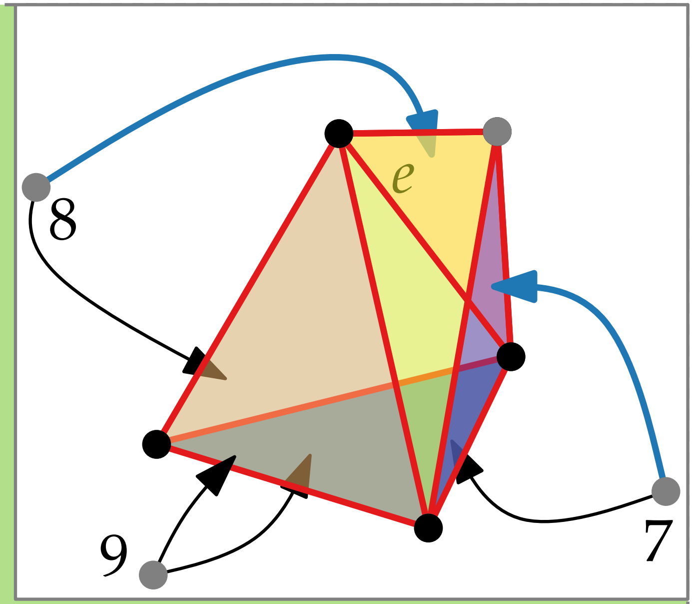
            $P(e) \leftarrow P_{\mathrm{conflict}}(f_1) \cup P_{\mathrm{conflict}}(f_2)$

            foreach $p \in P(e)$ **do**

                if f is visible from p **then** add edge (p, f) to G

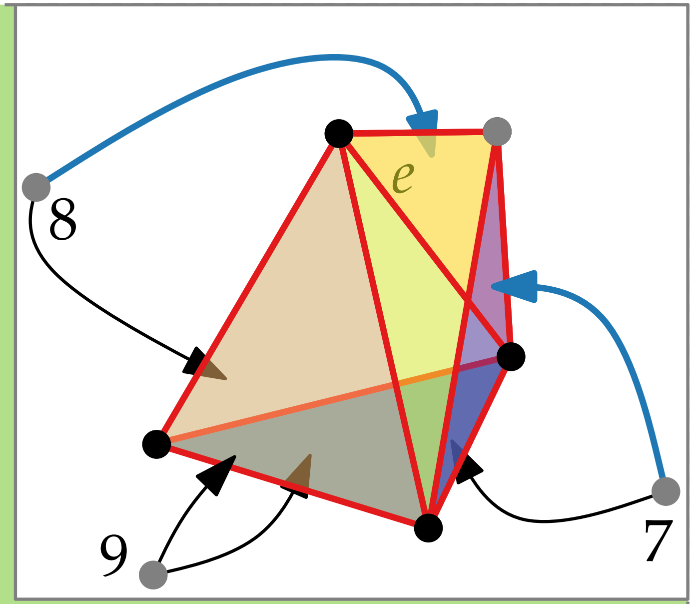
        delete vtc $\{p_r\} \cup F_{\mathrm{conflict}}(p_r)$ from G

return C



Worst-case running time:

Rand3DConvexHull($P \subset \mathbb{R}^3$)



pick non-coplanar set $P' = \{p_1, \ldots, p_4\} \subseteq P$

$C \leftarrow \mathrm{CH}(P')$

compute rand. perm. $(p_5, \ldots, p_n)$ of $P \setminus P'$

initialize conflict graph G

for $r = 5$ **to** n **do**

    if $F_{\text{conflict}}(p_r) \neq \emptyset$ **then** $\{ p_r \notin C \}$

        delete all facets in $F_{\text{conflict}}(p_r)$ from C

        $\mathcal{L} \leftarrow$ list of horizon edges visible from p_r

        foreach $e \in \mathcal{L}$ **do**

            $f \leftarrow C$.create_facet(e, p_r); create vtx for f in G

            $(f_1, f_2) \leftarrow$ previously_incident$_C(e)$

            $P(e) \leftarrow P_{\text{conflict}}(f_1) \cup P_{\text{conflict}}(f_2)$

            foreach $p \in P(e)$ **do**

                if f is visible from p **then** add edge (p, f) to G

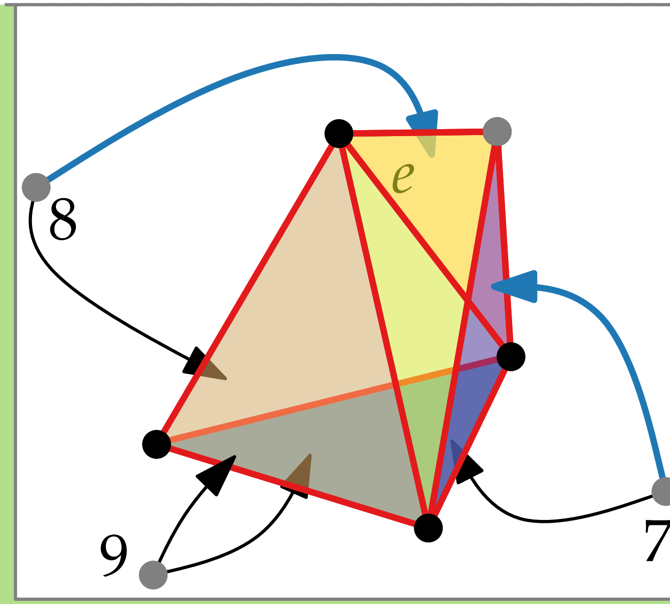
        delete vtc $\{p_r\} \cup F_{\text{conflict}}(p_r)$ from G

return C

Worst-case running time:

Rand3DConvexHull($P \subset \mathbb{R}^3$)

```
pick non-coplanar set P′ = {p_1, ..., p_4} ⊆ P
C ← CH(P′)
compute rand. perm. (p_5, ..., p_n) of P \ P′
initialize conflict graph G
for r = 5 to n do
    if F_conflict(p_r) ≠ ∅ then   { p_r ∉ C }
        delete all facets in F_conflict(p_r) from C
        ℒ ← list of horizon edges visible from p_r
        foreach e ∈ ℒ do
            f ← C.create_facet(e, p_r); create vtx for f in G
            (f_1, f_2) ← previously_incident_C(e)
            P(e) ← P_conflict(f_1) ∪ P_conflict(f_2)
            foreach p ∈ P(e) do
                if f is visible from p then add edge (p, f) to G
        delete vtc {p_r} ∪ F_conflict(p_r) from G
return C
```



Worst-case running time: $O(n^3)$

Computational Geometry

Lecture 9:
Convex Hulls in 3D
or
Mixing More Things

Part III:
Analysis

Philipp Kindermann

Winter Semester 2020

Analysis

Idea. Bound expected *structural change*

Analysis

Idea. Bound expected *structural change*,
that is, the total #facets created by the algorithm.

Analysis

Idea. Bound expected *structural change*,
that is, the total #facets created by the algorithm.

Lemma. The expected #facets created is at most $6n - 20$.

Analysis

Idea. Bound expected *structural change*,
that is, the total #facets created by the algorithm.

Lemma. The expected #facets created is at most $6n - 20$.

Proof. $E[\#\text{facets created}] =$

Analysis

Idea. Bound expected *structural change*,
that is, the total #facets created by the algorithm.

Lemma. The expected #facets created is at most $6n - 20$.

Proof. $E[\text{\#facets created}] =$
$= 4 + \sum_{r=5}^{n} E[\text{\#facets incident to } p_r \text{ in } \mathrm{CH}(P_r)]$

Analysis

Idea. Bound expected *structural change*, that is, the total #facets created by the algorithm.

Lemma. The expected #facets created is at most $6n - 20$.

Proof. $E[\#\text{facets created}] =$

$= 4 + \sum_{r=5}^{n} E[\#\text{facets incident to } p_r \text{ in } \mathrm{CH}(P_r)]$

Analysis

Idea. Bound expected *structural change*, that is, the total #facets created by the algorithm.

Lemma. The expected #facets created is at most $6n - 20$.

#edges

Proof. $E[\text{\#facets created}] =$

$= 4 + \sum_{r=5}^{n} E[\text{\#facets incident to } p_r \text{ in } \mathrm{CH}(P_r)]$

Analysis

Idea. Bound expected *structural change*, that is, the total #facets created by the algorithm.

Lemma. The expected #facets created is at most $6n - 20$.

Proof. $E[\#\text{facets created}] =$

$$= 4 + \sum_{r=5}^{n} E[\underbrace{\#\text{facets incident to } p_r \text{ in } \mathrm{CH}(P_r)}_{\deg(p_r, \mathrm{CH}(P_r))}]$$

#edges

Analysis

Idea. Bound expected *structural change*,
that is, the total #facets created by the algorithm.

Lemma. The expected #facets created is at most $6n - 20$.

Proof. $E[\#\text{facets created}] =$

$$= 4 + \sum_{r=5}^{n} E\underbrace{[\overset{\#\text{edges}}{\#\text{facets}} \text{ incident to } p_r \text{ in } \mathrm{CH}(P_r)]}_{\deg(p_r, \mathrm{CH}(P_r))}$$

For $r > 4$:

$E[\deg(p_r, \mathrm{CH}(P_r))] =$

Analysis

Idea. Bound expected *structural change*, that is, the total #facets created by the algorithm.

Lemma. The expected #facets created is at most $6n - 20$.

Proof. $E[\text{\#facets created}] =$

$$= 4 + \sum_{r=5}^{n} E[\underbrace{\text{\#facets incident to } p_r \text{ in } \mathrm{CH}(P_r)}_{\deg(p_r, \mathrm{CH}(P_r))}]$$

#edges

For $r > 4$:

$$E[\deg(p_r, \mathrm{CH}(P_r))] = \frac{1}{r-4} \sum_{i=5}^{r} \deg(p_i, \mathrm{CH}(P_r))$$

Analysis

Idea. Bound expected *structural change*, that is, the total #facets created by the algorithm.

Lemma. The expected #facets created is at most $6n - 20$.

#edges

Proof. $E[\#\text{facets created}] =$

$$= 4 + \sum_{r=5}^{n} E\underbrace{[\#\text{facets incident to } p_r \text{ in } \mathrm{CH}(P_r)]}_{\deg(p_r, \mathrm{CH}(P_r))}$$

For $r > 4$:

$$E[\deg(p_r, \mathrm{CH}(P_r))] = \frac{1}{r-4} \sum_{i=5}^{r} \deg(p_i, \mathrm{CH}(P_r))$$

$$\leq \frac{1}{r-4} \left[\left(\sum_{i=1}^{r} \deg(p_i)\right) - 12\right]$$

Analysis

Idea. Bound expected *structural change*,
that is, the total #facets created by the algorithm.

Lemma. The expected #facets created is at most $6n - 20$.

#edges

Proof. $E[\#\text{facets created}] =$

$$= 4 + \sum_{r=5}^{n} E\underbrace{[\#\text{facets incident to } p_r \text{ in } \mathrm{CH}(P_r)]}_{\deg(p_r,\, \mathrm{CH}(P_r))}$$

For $r > 4$:

$$E[\deg(p_r, \mathrm{CH}(P_r))] = \frac{1}{r-4} \sum_{i=5}^{r} \deg(p_i, \mathrm{CH}(P_r))$$

$$\leq \frac{1}{r-4} \left[\underbrace{\left(\sum_{i=1}^{r} \deg(p_i)\right)}_{} - 12\right]$$

Analysis

Idea. Bound expected *structural change*,
that is, the total #facets created by the algorithm.

Lemma. The expected #facets created is at most $6n - 20$.

#edges

Proof. $E[\#\text{facets created}] =$

$$= 4 + \sum_{r=5}^{n} E[\underbrace{\#\text{facets incident to } p_r \text{ in } \mathrm{CH}(P_r)}_{\deg(p_r,\, \mathrm{CH}(P_r))}]$$

For $r > 4$:

$$E[\deg(p_r, \mathrm{CH}(P_r))] = \frac{1}{r-4} \sum_{i=5}^{r} \deg(p_i, \mathrm{CH}(P_r))$$

$$\leq \frac{1}{r-4} \left[\underbrace{\left(\sum_{i=1}^{r} \deg(p_i) \right)}_{2 \cdot \#\text{ edges of } \mathrm{CH}(P_r)} - 12 \right]$$

Analysis

Idea. Bound expected *structural change*,
that is, the total #facets created by the algorithm.

Lemma. The expected #facets created is at most $6n - 20$.

Proof. $E[\#\text{facets created}] =$

$$= 4 + \sum_{r=5}^{n} E\underbrace{[\#\text{facets incident to } p_r \text{ in } \mathrm{CH}(P_r)]}_{\deg(p_r, \mathrm{CH}(P_r))}$$

#edges

For $r > 4$:

$$E[\deg(p_r, \mathrm{CH}(P_r))] = \frac{1}{r-4} \sum_{i=5}^{r} \deg(p_i, \mathrm{CH}(P_r))$$

$$\leq \frac{1}{r-4} \left[\underbrace{\left(\sum_{i=1}^{r} \deg(p_i) \right)}_{2 \cdot \#\text{ edges of } \mathrm{CH}(P_r)} - 12 \right]$$

$$\leq \frac{1}{r-4} \left[2 \cdot (3r - 6) - 12 \right]$$

Analysis

Idea. Bound expected *structural change*, that is, the total #facets created by the algorithm.

Lemma. The expected #facets created is at most $6n - 20$.

Proof. $E[\#\text{facets created}] =$

#edges

$$= 4 + \sum_{r=5}^{n} E[\underbrace{\#\text{facets incident to } p_r \text{ in } \mathrm{CH}(P_r)}_{\deg(p_r, \mathrm{CH}(P_r))}]$$

For $r > 4$:

$$E[\deg(p_r, \mathrm{CH}(P_r))] = \frac{1}{r-4} \sum_{i=5}^{r} \deg(p_i, \mathrm{CH}(P_r))$$

$$\leq \frac{1}{r-4} [\underbrace{(\textstyle\sum_{i=1}^{r} \deg(p_i))}_{2 \cdot \#\text{ edges of } \mathrm{CH}(P_r)} - 12]$$

$$\leq \frac{1}{r-4} [2 \cdot (3r - 6) - 12] \leq 6$$

Analysis

Idea. Bound expected *structural change*, that is, the total #facets created by the algorithm.

Lemma. The expected #facets created is at most $6n - 20$.

#edges

Proof. $E[\#\text{facets created}] =$

$$= 4 + \sum_{r=5}^{n} E[\underbrace{\#\text{facets incident to } p_r \text{ in } \mathrm{CH}(P_r)}_{\deg(p_r, \mathrm{CH}(P_r))}] \leq$$

For $r > 4$:

$$E[\deg(p_r, \mathrm{CH}(P_r))] = \frac{1}{r-4} \sum_{i=5}^{r} \deg(p_i, \mathrm{CH}(P_r))$$

$$\leq \frac{1}{r-4} [(\underbrace{\textstyle\sum_{i=1}^{r} \deg(p_i)}_{2 \cdot \#\text{ edges of } \mathrm{CH}(P_r)}) - 12]$$

$$\leq \frac{1}{r-4} [2 \cdot (3r - 6) - 12] \leq 6$$

Analysis

Idea. Bound expected *structural change*, that is, the total #facets created by the algorithm.

Lemma. The expected #facets created is at most $6n - 20$.

#edges

Proof. $E[\#\text{facets created}] =$

$$= 4 + \sum_{r=5}^{n} E\underbrace{[\#\text{facets incident to } p_r \text{ in } \mathrm{CH}(P_r)]}_{\deg(p_r,\, \mathrm{CH}(P_r))} \le 6n - 20$$

For $r > 4$:

$$E[\deg(p_r, \mathrm{CH}(P_r))] = \frac{1}{r-4} \sum_{i=5}^{r} \deg(p_i, \mathrm{CH}(P_r))$$

$$\le \frac{1}{r-4} [\underbrace{(\textstyle\sum_{i=1}^{r} \deg(p_i))}_{2 \cdot \#\text{ edges of } \mathrm{CH}(P_r)} - 12]$$

$$\le \frac{1}{r-4} [2 \cdot (3r - 6) - 12] \le 6$$

Running Time

Theorem. The convex hull of a set of n pts in $\mathbb{R}^3$ can be computed in $O(\qquad)$ expected time.

Rand3DConvexHull($P \subset \mathbb{R}^3$)
- pick non-coplanar set $P' = \{p_1, \ldots, p_4\} \subseteq P$
- $C \leftarrow \mathrm{CH}(P')$
- compute rand. perm. $(p_5, \ldots, p_n)$ of $P \setminus P'$
- initialize conflict graph G
- **for** $r = 5$ **to** n **do**
  - **if** $F_{\mathrm{conflict}}(p_r) \neq \emptyset$ **then**
    - delete all facets in $F_{\mathrm{conflict}}(p_r)$ from C
    - $\mathcal{L} \leftarrow$ list of horizon edges visible from p_r
    - **foreach** $e \in \mathcal{L}$ **do**
      - $f \leftarrow C$.create_facet(e, p_r); create vtx for f in G
      - $(f_1, f_2) \leftarrow$ previously_incident$_C(e)$
      - $P(e) \leftarrow P_{\mathrm{conflict}}(f_1) \cup P_{\mathrm{conflict}}(f_2)$
      - **foreach** $p \in P(e)$ **do**
        - **if** f visible from p **then** add edge (p, f) to G
    - delete vtc $\{p_r\} \cup F_{\mathrm{conflict}}(p_r)$ from G
- **return** C

Running Time

Theorem. The convex hull of a set of n pts in $\mathbb{R}^3$ can be computed in $O(\qquad\qquad)$ expected time.

```
Rand3DConvexHull(P ⊂ ℝ³)
  pick non-coplanar set P′ = {p_1, ..., p_4} ⊆ P
  C ← CH(P′)
  compute rand. perm. (p_5, ..., p_n) of P \ P′
  initialize conflict graph G
  for r = 5 to n do
      if F_conflict(p_r) ≠ ∅ then
          delete all facets in F_conflict(p_r) from C
          ℒ ← list of horizon edges visible from p_r
          foreach e ∈ ℒ do
              f ← C.create_facet(e, p_r); create vtx for f in G
              (f_1, f_2) ← previously_incident_C(e)
              P(e) ← P_conflict(f_1) ∪ P_conflict(f_2)
              foreach p ∈ P(e) do
                  if f visible from p then add edge (p, f) to G
          delete vtc {p_r} ∪ F_conflict(p_r) from G
  return C
```

Running Time

Theorem. The convex hull of a set of n pts in $\mathbb{R}^3$ can be computed in $O(\qquad)$ expected time.

$O(n)$ time

```
Rand3DConvexHull(P ⊂ ℝ³)
    pick non-coplanar set P′ = {p_1, ..., p_4} ⊆ P
    C ← CH(P′)
    compute rand. perm. (p_5, ..., p_n) of P \ P′
    initialize conflict graph G
    for r = 5 to n do
        if F_conflict(p_r) ≠ ∅ then
            delete all facets in F_conflict(p_r) from C
            ℒ ← list of horizon edges visible from p_r
            foreach e ∈ ℒ do
                f ← C.create_facet(e, p_r); create vtx for f in G
                (f_1, f_2) ← previously_incident_C(e)
                P(e) ← P_conflict(f_1) ∪ P_conflict(f_2)
                foreach p ∈ P(e) do
                    if f visible from p then add edge (p, f) to G
            delete vtc {p_r} ∪ F_conflict(p_r) from G
    return C
```

Running Time

Theorem. The convex hull of a set of n pts in $\mathbb{R}^3$ can be computed in $O(\qquad)$ expected time.

$O(n)$ time

Rand3DConvexHull($P \subset \mathbb{R}^3$)
- pick non-coplanar set $P' = \{p_1, \dots, p_4\} \subseteq P$
- $C \leftarrow \mathrm{CH}(P')$
- compute rand. perm. $(p_5, \dots, p_n)$ of $P \setminus P'$
- initialize conflict graph G

for $r = 5$ **to** n **do**
- **if** $F_{\text{conflict}}(p_r) \neq \emptyset$ **then**
  - delete all facets in $F_{\text{conflict}}(p_r)$ from C
  - $\mathcal{L} \leftarrow$ list of horizon edges visible from p_r
  - **foreach** $e \in \mathcal{L}$ **do**
    - $f \leftarrow C$.create_facet(e, p_r); create vtx for f in G
    - $(f_1, f_2) \leftarrow$ previously_incident$_C(e)$
    - $P(e) \leftarrow P_{\text{conflict}}(f_1) \cup P_{\text{conflict}}(f_2)$
    - **foreach** $p \in P(e)$ **do**
      - **if** f visible from p **then** add edge (p, f) to G
  - delete vtc $\{p_r\} \cup F_{\text{conflict}}(p_r)$ from G

return C

Stage r of for-loop (w/o foreach loop)

Running Time

Theorem. The convex hull of a set of n pts in $\mathbb{R}^3$ can be computed in $O(\qquad)$ expected time.

$O(n)$ time

Rand3DConvexHull($P \subset \mathbb{R}^3$)

```
pick non-coplanar set P' = {p_1, ..., p_4} ⊆ P
C ← CH(P')
compute rand. perm. (p_5, ..., p_n) of P \ P'
initialize conflict graph G
for r = 5 to n do
    if F_conflict(p_r) ≠ ∅ then
        delete all facets in F_conflict(p_r) from C
        ℒ ← list of horizon edges visible from p_r
        foreach e ∈ ℒ do
            f ← C.create_facet(e, p_r); create vtx for f in G
            (f_1, f_2) ← previously_incident_C(e)
            P(e) ← P_conflict(f_1) ∪ P_conflict(f_2)
            foreach p ∈ P(e) do
                if f visible from p then add edge (p, f) to G
        delete vtc {p_r} ∪ F_conflict(p_r) from G
return C
```

Stage r of for-loop (w/o foreach loop) takes time $O(|F_{\text{conflict}}(p_r)|) =$

Running Time

Theorem. The convex hull of a set of n pts in $\mathbb{R}^3$ can be computed in $O(\qquad)$ expected time.

$O(n)$ time

Rand3DConvexHull($P \subset \mathbb{R}^3$)
- pick non-coplanar set $P' = \{p_1, \dots, p_4\} \subseteq P$
- $C \leftarrow \mathrm{CH}(P')$
- compute rand. perm. $(p_5, \dots, p_n)$ of $P \setminus P'$
- initialize conflict graph G

for $r = 5$ **to** n **do**
- **if** $F_{\text{conflict}}(p_r) \neq \emptyset$ **then**
  - delete all facets in $F_{\text{conflict}}(p_r)$ from C
  - $\mathcal{L} \leftarrow$ list of horizon edges visible from p_r
  - **foreach** $e \in \mathcal{L}$ **do**
    - $f \leftarrow C$.create_facet(e, p_r); create vtx for f in G
    - $(f_1, f_2) \leftarrow$ previously_incident$_C(e)$
    - $P(e) \leftarrow P_{\text{conflict}}(f_1) \cup P_{\text{conflict}}(f_2)$
    - **foreach** $p \in P(e)$ **do**
      - **if** f visible from p **then** add edge (p, f) to G
  - delete vtc $\{p_r\} \cup F_{\text{conflict}}(p_r)$ from G

return C

Stage r of for-loop (w/o foreach loop) takes time $O(|F_{\text{conflict}}(p_r)|) = O(\#\text{facets del. when adding } p_r)$

Running Time

Theorem. The convex hull of a set of n pts in $\mathbb{R}^3$ can be computed in $O(\qquad)$ expected time.

$O(n)$ time

```
Rand3DConvexHull(P ⊂ ℝ³)
  pick non-coplanar set P′ = {p_1, ..., p_4} ⊆ P
  C ← CH(P′)
  compute rand. perm. (p_5, ..., p_n) of P \ P′
  initialize conflict graph G
  for r = 5 to n do
    if F_conflict(p_r) ≠ ∅ then
      delete all facets in F_conflict(p_r) from C
      ℒ ← list of horizon edges visible from p_r
      foreach e ∈ ℒ do
        f ← C.create_facet(e, p_r); create vtx for f in G
        (f_1, f_2) ← previously_incident_C(e)
        P(e) ← P_conflict(f_1) ∪ P_conflict(f_2)
        foreach p ∈ P(e) do
          if f visible from p then add edge (p, f) to G
      delete vtc {p_r} ∪ F_conflict(p_r) from G
  return C
```

Stage r of for-loop (w/o foreach loop) takes time $O(|F_{\text{conflict}}(p_r)|) = O(\#\text{facets del. when adding } p_r)$

This part of for-loop in total:

Running Time

Theorem. The convex hull of a set of n pts in $\mathbb{R}^3$ can be computed in $O(\qquad)$ expected time.

$O(n)$ time

```
Rand3DConvexHull(P ⊂ ℝ³)
  pick non-coplanar set P′ = {p_1, ..., p_4} ⊆ P
  C ← CH(P′)
  compute rand. perm. (p_5, ..., p_n) of P \ P′
  initialize conflict graph G
  for r = 5 to n do
    if F_conflict(p_r) ≠ ∅ then
      delete all facets in F_conflict(p_r) from C
      ℒ ← list of horizon edges visible from p_r
      foreach e ∈ ℒ do
        f ← C.create_facet(e, p_r); create vtx for f in G
        (f_1, f_2) ← previously_incident_C(e)
        P(e) ← P_conflict(f_1) ∪ P_conflict(f_2)
        foreach p ∈ P(e) do
          if f visible from p then add edge (p, f) to G
      delete vtc {p_r} ∪ F_conflict(p_r) from G
  return C
```

Stage r of for-loop (w/o foreach loop) takes time $O(|F_{\text{conflict}}(p_r)|) = O(\#\text{facets del. when adding } p_r)$

This part of for-loop in total: $E[\#\text{facets deleted}] =$

Running Time

Theorem. The convex hull of a set of n pts in $\mathbb{R}^3$ can be computed in $O(\quad\quad)$ expected time.

$O(n)$ time

```
Rand3DConvexHull(P ⊂ ℝ³)
  pick non-coplanar set P' = {p_1, ..., p_4} ⊆ P
  C ← CH(P')
  compute rand. perm. (p_5, ..., p_n) of P \ P'
  initialize conflict graph G
  for r = 5 to n do
      if F_conflict(p_r) ≠ ∅ then
          delete all facets in F_conflict(p_r) from C
          ℒ ← list of horizon edges visible from p_r
          foreach e ∈ ℒ do
              f ← C.create_facet(e, p_r); create vtx for f in G
              (f_1, f_2) ← previously_incident_C(e)
              P(e) ← P_conflict(f_1) ∪ P_conflict(f_2)
              foreach p ∈ P(e) do
                  if f visible from p then add edge (p, f) to G
          delete vtc {p_r} ∪ F_conflict(p_r) from G
  return C
```

Stage r of for-loop (w/o foreach loop) takes time $O(|F_{\text{conflict}}(p_r)|) = O(\#\text{facets del. when adding } p_r)$

This part of for-loop in total:
$E[\#\text{facets deleted}] =$
$\leq E[\#\text{facets created}] =$

Running Time

Theorem. The convex hull of a set of n pts in $\mathbb{R}^3$ can be computed in $O(\qquad)$ expected time.

$O(n)$ time

Rand3DConvexHull($P \subset \mathbb{R}^3$)

```
pick non-coplanar set P' = {p_1, ..., p_4} ⊆ P
C ← CH(P')
compute rand. perm. (p_5, ..., p_n) of P \ P'
initialize conflict graph G
for r = 5 to n do
    if F_conflict(p_r) ≠ ∅ then
        delete all facets in F_conflict(p_r) from C
        ℒ ← list of horizon edges visible from p_r
        foreach e ∈ ℒ do
            f ← C.create_facet(e, p_r); create vtx for f in G
            (f_1, f_2) ← previously_incident_C(e)
            P(e) ← P_conflict(f_1) ∪ P_conflict(f_2)
            foreach p ∈ P(e) do
                if f visible from p then add edge (p, f) to G
        delete vtc {p_r} ∪ F_conflict(p_r) from G
return C
```

Stage r of for-loop (w/o foreach loop) takes time $O(|F_{\text{conflict}}(p_r)|) = O(\text{\#facets del. when adding } p_r)$

This part of for-loop in total:

$E[\text{\#facets deleted}] =$

$\leq E[\text{\#facets created}] =$

Lemma

Running Time

Theorem. The convex hull of a set of n pts in $\mathbb{R}^3$ can be computed in $O(\qquad)$ expected time.

$O(n)$ time

Rand3DConvexHull($P \subset \mathbb{R}^3$)
- pick non-coplanar set $P' = \{p_1, \dots, p_4\} \subseteq P$
- $C \leftarrow \mathrm{CH}(P')$
- compute rand. perm. $(p_5, \dots, p_n)$ of $P \setminus P'$
- initialize conflict graph G

for $r = 5$ **to** n **do**
- **if** $F_{\mathrm{conflict}}(p_r) \neq \emptyset$ **then**
  - delete all facets in $F_{\mathrm{conflict}}(p_r)$ from C
  - $\mathcal{L} \leftarrow$ list of horizon edges visible from p_r
  - **foreach** $e \in \mathcal{L}$ **do**
    - $f \leftarrow C$.create_facet(e, p_r); create vtx for f in G
    - $(f_1, f_2) \leftarrow$ previously_incident$_C(e)$
    - $P(e) \leftarrow P_{\mathrm{conflict}}(f_1) \cup P_{\mathrm{conflict}}(f_2)$
    - **foreach** $p \in P(e)$ **do**
      - **if** f visible from p **then** add edge (p, f) to G
  - delete vtc $\{p_r\} \cup F_{\mathrm{conflict}}(p_r)$ from G

return C

Stage r of for-loop (w/o foreach loop) takes time $O(|F_{\mathrm{conflict}}(p_r)|) =$ $O(\#\text{facets del. when adding } p_r)$

This part of for-loop in total:
$E[\#\text{facets deleted}] =$
$\leq E[\#\text{facets created}] = O(n)$.

Lemma

Running Time

Theorem. The convex hull of a set of n pts in $\mathbb{R}^3$ can be computed in $O(\qquad)$ expected time.

$O(n)$ time

Rand3DConvexHull($P \subset \mathbb{R}^3$)
- pick non-coplanar set $P' = \{p_1, \dots, p_4\} \subseteq P$
- $C \leftarrow \mathrm{CH}(P')$
- compute rand. perm. $(p_5, \dots, p_n)$ of $P \setminus P'$
- initialize conflict graph G

for $r = 5$ **to** n **do**
- **if** $F_{\text{conflict}}(p_r) \neq \emptyset$ **then**
  - delete all facets in $F_{\text{conflict}}(p_r)$ from C
  - $\mathcal{L} \leftarrow$ list of horizon edges visible from p_r
  - **foreach** $e \in \mathcal{L}$ **do**
    - $f \leftarrow C$.create_facet(e, p_r); create vtx for f in G
    - $(f_1, f_2) \leftarrow$ previously_incident$_C(e)$
    - $P(e) \leftarrow P_{\text{conflict}}(f_1) \cup P_{\text{conflict}}(f_2)$
    - **foreach** $p \in P(e)$ **do**
      - **if** f visible from p **then** add edge (p, f) to G
  - delete vtc $\{p_r\} \cup F_{\text{conflict}}(p_r)$ from G

return C

Stage r of for-loop (w/o foreach loop) takes time $O(|F_{\text{conflict}}(p_r)|) = O(\#\text{facets del. when adding } p_r)$

This part of for-loop in total:
$E[\#\text{facets deleted}] =$
$\leq E[\#\text{facets created}] = O(n)$.
Lemma

Outer foreach-loop:

Running Time

Theorem. The convex hull of a set of n pts in $\mathbb{R}^3$ can be computed in $O(\quad\quad)$ expected time.

$O(n)$ time

```
Rand3DConvexHull(P ⊂ ℝ³)
  pick non-coplanar set P′ = {p_1, ..., p_4} ⊆ P
  C ← CH(P′)
  compute rand. perm. (p_5, ..., p_n) of P \ P′
  initialize conflict graph G
  for r = 5 to n do
      if F_conflict(p_r) ≠ ∅ then
          delete all facets in F_conflict(p_r) from C
          ℒ ← list of horizon edges visible from p_r
          foreach e ∈ ℒ do
              f ← C.create_facet(e, p_r); create vtx for f in G
              (f_1, f_2) ← previously_incident_C(e)
              P(e) ← P_conflict(f_1) ∪ P_conflict(f_2)
              foreach p ∈ P(e) do
                  if f visible from p then add edge (p, f) to G
          delete vtc {p_r} ∪ F_conflict(p_r) from G
  return C
```

Stage r of for-loop (w/o foreach loop) takes time $O(|F_{\text{conflict}}(p_r)|) = O(\#\text{facets del. when adding } p_r)$

This part of for-loop in total:
$E[\#\text{facets deleted}] =$
$\le E[\#\text{facets created}] = O(n)$.
Lemma

Outer foreach-loop:
– in stage r: $O(\sum_{e \in \mathcal{L}} |P(e)|)$

Running Time

Theorem. The convex hull of a set of n pts in $\mathbb{R}^3$ can be computed in $O(\qquad)$ expected time.

$O(n)$ time

Rand3DConvexHull($P \subset \mathbb{R}^3$)
pick non-coplanar set $P' = \{p_1, \dots, p_4\} \subseteq P$
$C \leftarrow \mathrm{CH}(P')$
compute rand. perm. $(p_5, \dots, p_n)$ of $P \setminus P'$
initialize conflict graph G
for $r = 5$ **to** n **do**
  if $F_{\text{conflict}}(p_r) \neq \emptyset$ **then**
    delete all facets in $F_{\text{conflict}}(p_r)$ from C
    $\mathcal{L} \leftarrow$ list of horizon edges visible from p_r
    foreach $e \in \mathcal{L}$ **do**
      $f \leftarrow C$.create_facet(e, p_r); create vtx for f in G
      $(f_1, f_2) \leftarrow$ previously_incident$_C(e)$
      $P(e) \leftarrow P_{\text{conflict}}(f_1) \cup P_{\text{conflict}}(f_2)$
      foreach $p \in P(e)$ **do**
        if f visible from p **then** add edge (p, f) to G
    delete vtc $\{p_r\} \cup F_{\text{conflict}}(p_r)$ from G
return C

Stage r of for-loop (w/o foreach loop) takes time $O(|F_{\text{conflict}}(p_r)|) =$ $O(\text{\#facets del. when adding } p_r)$

This part of for-loop in total:
$E[\text{\#facets deleted}] =$
$\leq E[\text{\#facets created}] = O(n)$.
Lemma

Outer foreach-loop:
– in stage r: $O(\sum_{e \in \mathcal{L}} |P(e)|)$
– in total:

Running Time

Theorem. The convex hull of a set of n pts in $\mathbb{R}^3$ can be computed in $O(\qquad)$ expected time.

$O(n)$ time

```
Rand3DConvexHull(P ⊂ ℝ³)
  pick non-coplanar set P′ = {p_1, ..., p_4} ⊆ P
  C ← CH(P′)
  compute rand. perm. (p_5, ..., p_n) of P \ P′
  initialize conflict graph G
  for r = 5 to n do
      if F_conflict(p_r) ≠ ∅ then
          delete all facets in F_conflict(p_r) from C
          ℒ ← list of horizon edges visible from p_r
          foreach e ∈ ℒ do
              f ← C.create_facet(e, p_r); create vtx for f in G
              (f_1, f_2) ← previously_incident_C(e)
              P(e) ← P_conflict(f_1) ∪ P_conflict(f_2)
              foreach p ∈ P(e) do
                  if f visible from p then add edge (p, f) to G
          delete vtc {p_r} ∪ F_conflict(p_r) from G
  return C
```

Stage r of for-loop (w/o foreach loop) takes time $O(|F_{\text{conflict}}(p_r)|) =$ $O(\#\text{facets del. when adding } p_r)$

This part of for-loop in total:
$E[\#\text{facets deleted}] =$
$\leq E[\#\text{facets created}] = O(n).$
Lemma

Outer foreach-loop:
– in stage r: $O(\sum_{e \in \mathcal{L}} |P(e)|)$
– in total:

$$O\left(\sum_{e \text{ on horizon at some time}} |P(e)|\right)$$

Running Time

Theorem. The convex hull of a set of n pts in $\mathbb{R}^3$ can be computed in $O(\qquad)$ expected time.

$O(n)$ time

```
Rand3DConvexHull(P ⊂ ℝ³)
  pick non-coplanar set P' = {p_1, ..., p_4} ⊆ P
  C ← CH(P')
  compute rand. perm. (p_5, ..., p_n) of P \ P'
  initialize conflict graph G
  for r = 5 to n do
    if F_conflict(p_r) ≠ ∅ then
      delete all facets in F_conflict(p_r) from C
      ℒ ← list of horizon edges visible from p_r
      foreach e ∈ ℒ do
        f ← C.create_facet(e, p_r); create vtx for f in G
        (f_1, f_2) ← previously_incident_C(e)
        P(e) ← P_conflict(f_1) ∪ P_conflict(f_2)
        foreach p ∈ P(e) do
          if f visible from p then add edge (p, f) to G
      delete vtc {p_r} ∪ F_conflict(p_r) from G
  return C
```

Stage r of for-loop (w/o foreach loop) takes time $O(|F_{\text{conflict}}(p_r)|) = O(\#\text{facets del. when adding } p_r)$

This part of for-loop in total:

$E[\#\text{facets deleted}] = \\ \leq E[\#\text{facets created}] \underset{\text{Lemma}}{=} O(n).$

Outer foreach-loop:

– in stage r: $O(\sum_{e \in \mathcal{L}} |P(e)|)$

– in total:

$$O\left(\sum_{e \text{ on horizon at some time}} |P(e)|\right) = O(n^2)$$

Running Time

Theorem. The convex hull of a set of n pts in $\mathbb{R}^3$ can be computed in $O(\qquad)$ expected time.

$O(n)$ time

```
Rand3DConvexHull(P ⊂ ℝ³)
  pick non-coplanar set P′={p_1,...,p_4}⊆P
  C ← CH(P′)
  compute rand. perm. (p_5,...,p_n) of P\P′
  initialize conflict graph G
  for r = 5 to n do
      if F_conflict(p_r) ≠ ∅ then
          delete all facets in F_conflict(p_r) from C
          ℒ ← list of horizon edges visible from p_r
          foreach e ∈ ℒ do
              f ← C.create_facet(e, p_r); create vtx for f in G
              (f_1, f_2) ← previously_incident_C(e)
              P(e) ← P_conflict(f_1) ∪ P_conflict(f_2)
              foreach p ∈ P(e) do
                  if f visible from p then add edge (p, f) to G
          delete vtc {p_r} ∪ F_conflict(p_r) from G
  return C
```

using *configuration spaces*, Section 9.5 [Comp. Geom A&A]

Stage r of for-loop (w/o foreach loop) takes time $O(|F_{\text{conflict}}(p_r)|) = O(\#\text{facets del. when adding } p_r)$

This part of for-loop in total: $E[\#\text{facets deleted}] = \leq E[\#\text{facets created}] = O(n)$.

Lemma

Outer foreach-loop:

– in stage r: $O(\sum_{e \in \mathcal{L}} |P(e)|)$

– in total:

$$O\left(\sum_{e \text{ on horizon at some time}} |P(e)|\right) = O(n^2)$$

Running Time

Theorem. The convex hull of a set of n pts in $\mathbb{R}^3$ can be computed in $O(\qquad)$ expected time.

$O(n)$ time

Rand3DConvexHull($P \subset \mathbb{R}^3$)
pick non-coplanar set $P' = \{p_1, \dots, p_4\} \subseteq P$
$C \leftarrow \mathrm{CH}(P')$
compute rand. perm. $(p_5, \dots, p_n)$ of $P \setminus P'$
initialize conflict graph G
for $r = 5$ **to** n **do**
  if $F_{\mathrm{conflict}}(p_r) \neq \emptyset$ **then**
    delete all facets in $F_{\mathrm{conflict}}(p_r)$ from C
    $\mathcal{L} \leftarrow$ list of horizon edges visible from p_r
    foreach $e \in \mathcal{L}$ **do**
      $f \leftarrow C$.create_facet(e, p_r); create vtx for f in G
      $(f_1, f_2) \leftarrow$ previously_incident$_C(e)$
      $P(e) \leftarrow P_{\mathrm{conflict}}(f_1) \cup P_{\mathrm{conflict}}(f_2)$
      foreach $p \in P(e)$ **do**
        if f visible from p **then** add edge (p, f) to G
    delete vtc $\{p_r\} \cup F_{\mathrm{conflict}}(p_r)$ from G
return C

using *configuration spaces*, Section 9.5 [Comp. Geom A&A]

Stage r of for-loop (w/o foreach loop) takes time $O(|F_{\mathrm{conflict}}(p_r)|) =$ $O(\#\text{facets del. when adding } p_r)$

This part of for-loop in total:
$E[\#\text{facets deleted}] =$
$\leq E[\#\text{facets created}] = O(n)$.
Lemma

Outer foreach-loop:
– in stage r: $O(\sum_{e \in \mathcal{L}} |P(e)|)$
– in total:

$$O\left(\sum_{e \text{ on horizon at some time}} |P(e)| \right)$$

$$= \cancel{O(n^2)}\ O(n \log n)$$

Running Time

Theorem. The convex hull of a set of n pts in $\mathbb{R}^3$ can be computed in $O(n \log n)$ expected time.

$O(n)$ time

Rand3DConvexHull($P \subset \mathbb{R}^3$)
- pick non-coplanar set $P' = \{p_1, \ldots, p_4\} \subseteq P$
- $C \leftarrow \mathrm{CH}(P')$
- compute rand. perm. $(p_5, \ldots, p_n)$ of $P \setminus P'$
- initialize conflict graph G

for $r = 5$ **to** n **do**
- **if** $F_{\mathrm{conflict}}(p_r) \neq \emptyset$ **then**
  - delete all facets in $F_{\mathrm{conflict}}(p_r)$ from C
  - $\mathcal{L} \leftarrow$ list of horizon edges visible from p_r
  - **foreach** $e \in \mathcal{L}$ **do**
    - $f \leftarrow C$.create_facet(e, p_r); create vtx for f in G
    - $(f_1, f_2) \leftarrow$ previously_incident$_C(e)$
    - $P(e) \leftarrow P_{\mathrm{conflict}}(f_1) \cup P_{\mathrm{conflict}}(f_2)$
    - **foreach** $p \in P(e)$ **do**
      - **if** f visible from p **then** add edge (p, f) to G
  - delete vtc $\{p_r\} \cup F_{\mathrm{conflict}}(p_r)$ from G

return C

using *configuration spaces*, Section 9.5 [Comp. Geom A&A]

Stage r of for-loop (w/o foreach loop) takes time $O(|F_{\mathrm{conflict}}(p_r)|) = O(\text{\#facets del. when adding } p_r)$

This part of for-loop in total:
$E[\text{\#facets deleted}] = \; \leq E[\text{\#facets created}] = O(n)$.

Lemma

Outer foreach-loop:
- in stage r: $O(\sum_{e \in \mathcal{L}} |P(e)|)$
- in total:

$$O\left(\sum_{e \text{ on horizon at some time}} |P(e)| \right)$$

$$= \cancel{O(n^2)}\; O(n \log n)$$

Running Time

$\mathbb{R}^d, d > 3$

Theorem. The convex hull of a set of n pts in $\mathbb{R}^3$ can be computed in $O(n \log n)$ expected time.

$O(n)$ time

Rand3DConvexHull($P \subset \mathbb{R}^3$)
pick non-coplanar set $P' = \{p_1, \dots, p_4\} \subseteq P$
$C \leftarrow \mathrm{CH}(P')$
compute rand. perm. $(p_5, \dots, p_n)$ of $P \setminus P'$
initialize conflict graph G
for $r = 5$ **to** n **do**
  if $F_{\text{conflict}}(p_r) \neq \emptyset$ **then**
    delete all facets in $F_{\text{conflict}}(p_r)$ from C
    $\mathcal{L} \leftarrow$ list of horizon edges visible from p_r
    foreach $e \in \mathcal{L}$ **do**
      $f \leftarrow C$.create_facet(e, p_r); create vtx for f in G
      $(f_1, f_2) \leftarrow$ previously_incident$_C(e)$
      $P(e) \leftarrow P_{\text{conflict}}(f_1) \cup P_{\text{conflict}}(f_2)$
      foreach $p \in P(e)$ **do**
        if f visible from p **then** add edge (p, f) to G
    delete vtc $\{p_r\} \cup F_{\text{conflict}}(p_r)$ from G
return C

using *configuration spaces*, Section 9.5 [Comp. Geom A&A]

Stage r of for-loop (w/o foreach loop) takes time $O(|F_{\text{conflict}}(p_r)|) = O(\#\text{facets del. when adding } p_r)$

This part of for-loop in total:
$E[\#\text{facets deleted}] = \leq E[\#\text{facets created}] = O(n)$.
Lemma

Outer foreach-loop:
– in stage r: $O(\sum_{e \in \mathcal{L}} |P(e)|)$
– in total:

$$O\left(\sum_{e \text{ on horizon at some time}} |P(e)|\right)$$

$$= \cancel{O(n^2)}\ O(n \log n)$$

Running Time

$\mathbb{R}^d, d > 3$

Theorem. The convex hull of a set of n pts in $\mathbb{R}^3$ can be computed in $O(n \log n)$ expected time.

$O(n^{\lfloor d/2 \rfloor})$

$O(n)$ time

```
Rand3DConvexHull(P ⊂ ℝ³)
  pick non-coplanar set P′ = {p_1, ..., p_4} ⊆ P
  C ← CH(P′)
  compute rand. perm. (p_5, ..., p_n) of P \ P′
  initialize conflict graph G
  for r = 5 to n do
    if F_conflict(p_r) ≠ ∅ then
      delete all facets in F_conflict(p_r) from C
      ℒ ← list of horizon edges visible from p_r
      foreach e ∈ ℒ do
        f ← C.create_facet(e, p_r); create vtx for f in G
        (f_1, f_2) ← previously_incident_C(e)
        P(e) ← P_conflict(f_1) ∪ P_conflict(f_2)
        foreach p ∈ P(e) do
          if f visible from p then add edge (p, f) to G
      delete vtc {p_r} ∪ F_conflict(p_r) from G
  return C
```

Stage r of for-loop (w/o foreach loop) takes time $O(|F_{\text{conflict}}(p_r)|) =$ $O(\#\text{facets del. when adding } p_r)$

This part of for-loop in total:
$E[\#\text{facets deleted}] =$
$\le E[\#\text{facets created}] = O(n)$.
Lemma

Outer foreach-loop:
– in stage r: $O(\sum_{e \in \mathcal{L}} |P(e)|)$
– in total:

$$O\left(\sum_{e \text{ on horizon at some time}} |P(e)|\right)$$

$$= \cancel{O(n^2)}\ O(n \log n)$$

using *configuration spaces*, Section 9.5 [Comp. Geom A&A]

Computational Geometry

Lecture 9:
Convex Hulls in 3D
or
Mixing More Things

Part IV:
Half-Space Intersections

Philipp Kindermann

Winter Semester 2020

Convex Hulls and Half-Space Intersections

Convex Hulls and Half-~~Space~~ Intersections
Plane

Convex Hulls and Half-~~Space~~ Plane Intersections

Define dualtity $\star$ between pts and (non-vertical) lines:

Convex Hulls and Half-~~Space~~ Intersections Plane

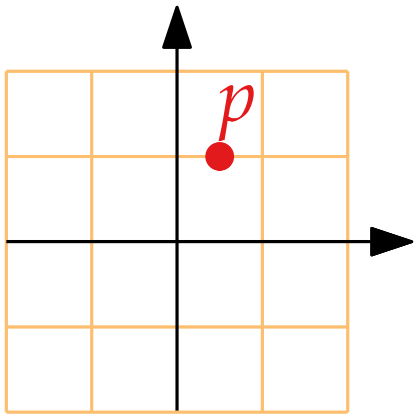
Define dualtity $\star$ between pts and (non-vertical) lines:

For $p = (p_x, p_y)$,

Convex Hulls and Half-~~Space~~ Plane Intersections

Define dualtity $\star$ between pts and (non-vertical) lines:

For $p = (p_x, p_y)$,

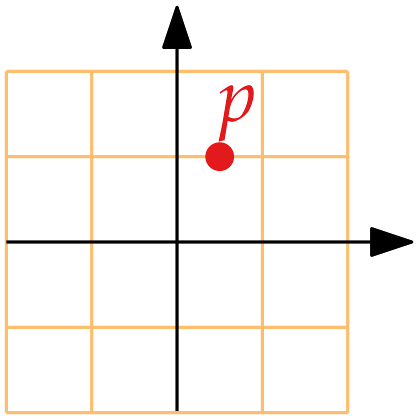


primal

Convex Hulls and Half-~~Space~~ Intersections Plane

Define dualtity $\star$ between pts and (non-vertical) lines:

For $p = (p_x, p_y)$, define the line $p^\star \colon y = p_x x - p_y$.

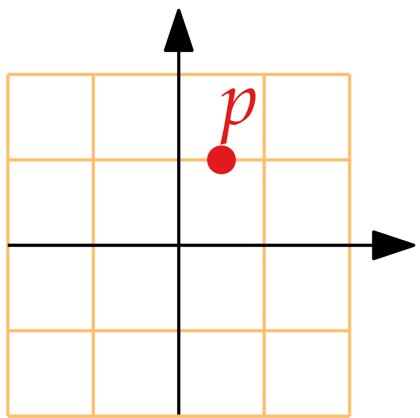


primal

Convex Hulls and Half-~~Space~~ Plane Intersections

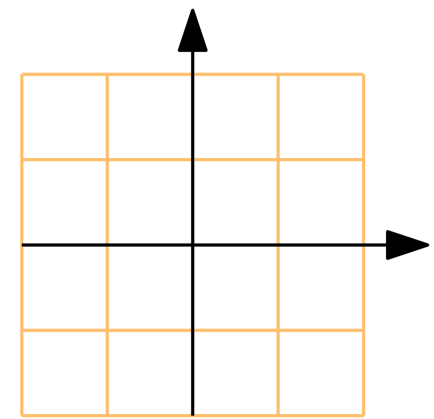
Define dualtity $\star$ between pts and (non-vertical) lines:

For $p = (p_x, p_y)$, define the line $p^\star \colon y = p_x x - p_y$.



primal

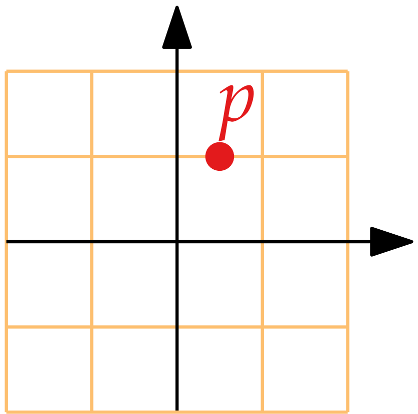
dual



Convex Hulls and Half-~~Space~~ Plane Intersections

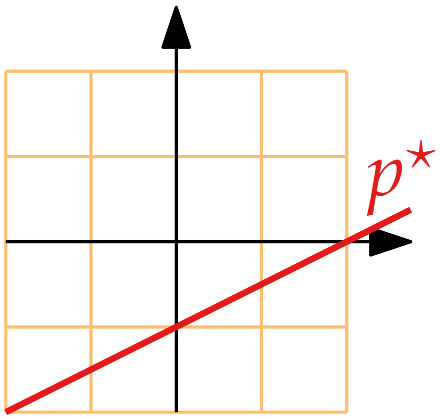
Define dualtity $\star$ between pts and (non-vertical) lines:

For $p = (p_x, p_y)$, define the line $p^\star \colon y = p_x x - p_y$.



primal

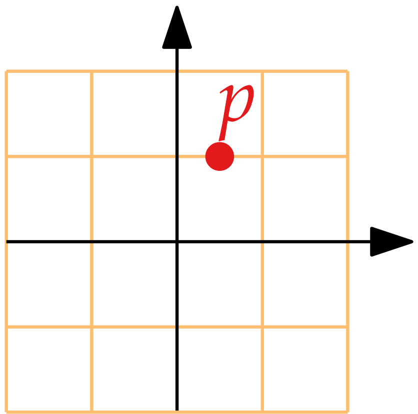
dual

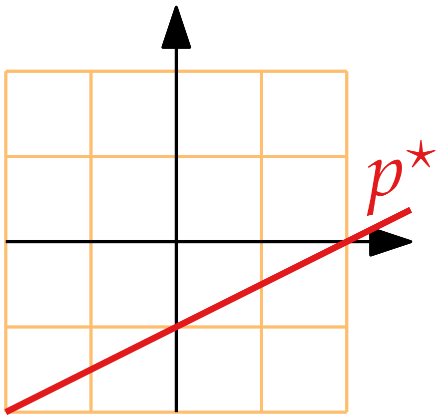


Convex Hulls and Half-~~Space~~ Plane Intersections

Define dualtity $\star$ between pts and (non-vertical) lines:

For $p = (p_x, p_y)$, define the line $p^\star : y = p_x x - p_y$.



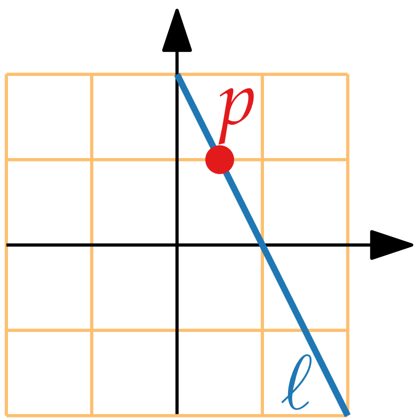


For $\ell : y = mx + b$,

Convex Hulls and Half-~~Space~~ Plane Intersections

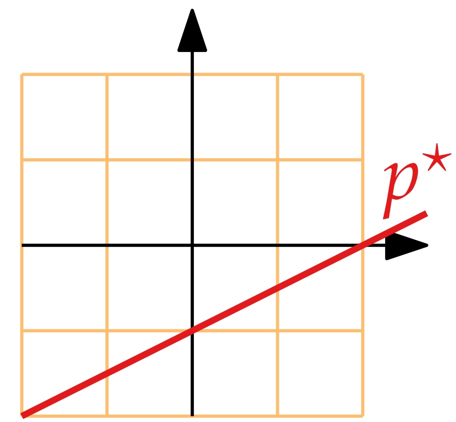
Define dualtity $\star$ between pts and (non-vertical) lines:

For $p = (p_x, p_y)$, define the line $p^\star \colon y = p_x x - p_y$.



primal

dual

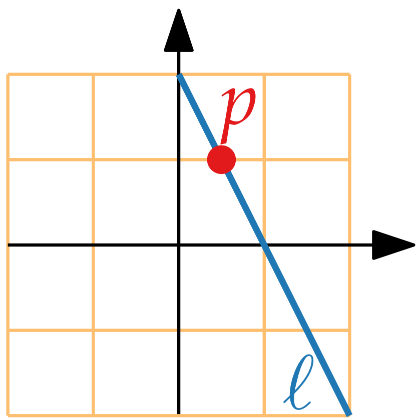


For $\ell \colon y = mx + b$,

Convex Hulls and Half-~~Space~~ Plane Intersections

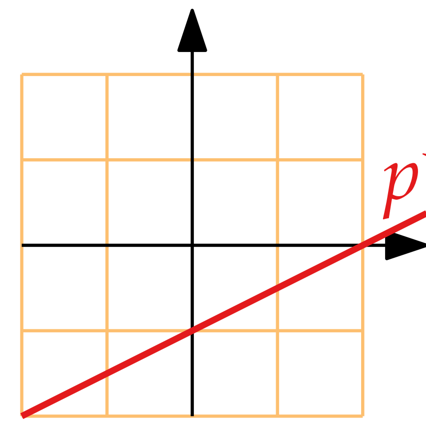
Define dualtity $\star$ between pts and (non-vertical) lines:

For $p = (p_x, p_y)$, define the line $p^\star \colon y = p_x x - p_y$.



primal

dual

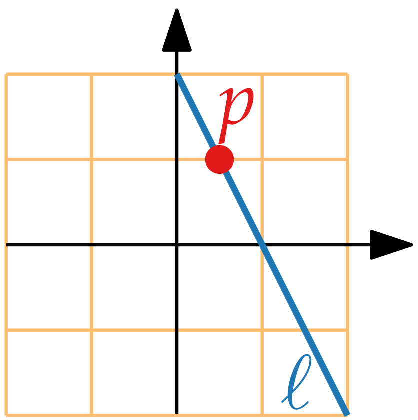


For $\ell \colon y = mx + b$, define $\ell^\star$ to be the pt q with $q^\star = \ell$

Convex Hulls and Half-~~Space~~ Plane Intersections

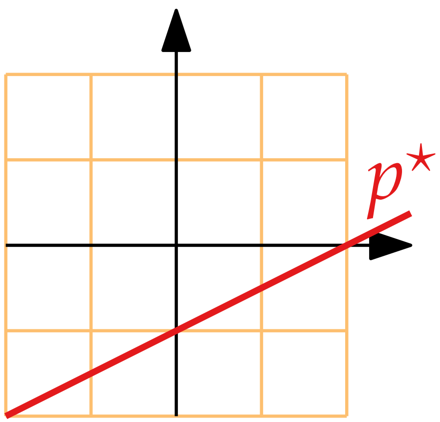
Define dualtity $\star$ between pts and (non-vertical) lines:

For $p = (p_x, p_y)$, define the line $p^\star \colon y = p_x x - p_y$.



primal

dual



For $\ell \colon y = mx + b$, define $\ell^\star$ to be the pt q with $q^\star = \ell$, that is, $\ell^\star = (m, -b)$.

Convex Hulls and Half-~~Space~~ Plane Intersections

Define dualtity $\star$ between pts and (non-vertical) lines:

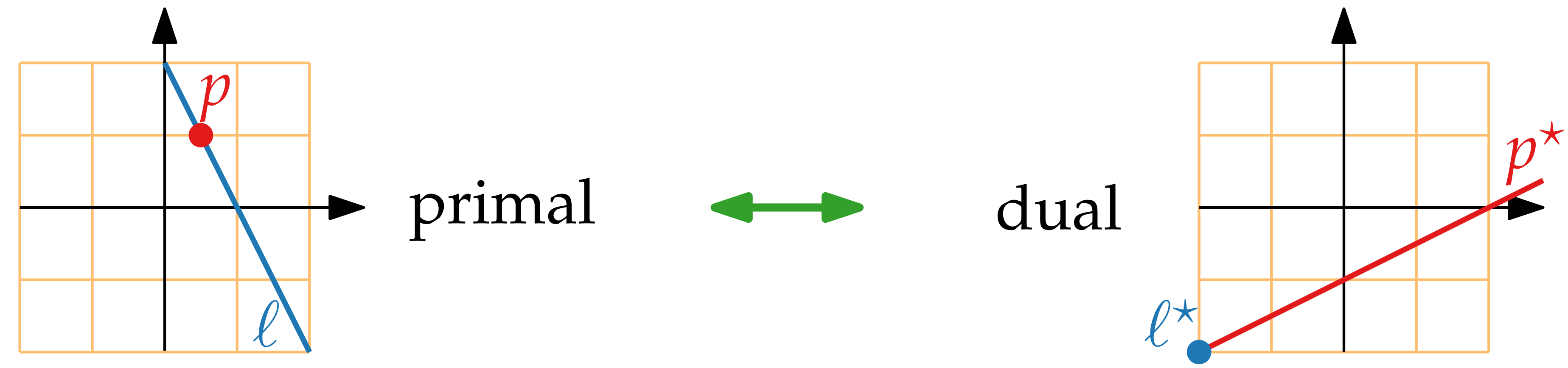
For $p = (p_x, p_y)$, define the line $p^\star \colon y = p_x x - p_y$.

For $\ell \colon y = mx + b$, define $\ell^\star$ to be the pt q with $q^\star = \ell$,
that is, $\ell^\star = (m, -b)$.

Convex Hulls and Half-~~Space~~ Plane Intersections

Define dualtity $\star$ between pts and (non-vertical) lines:

For $p = (p_x, p_y)$, define the line $p^\star\colon y = p_x x - p_y$.



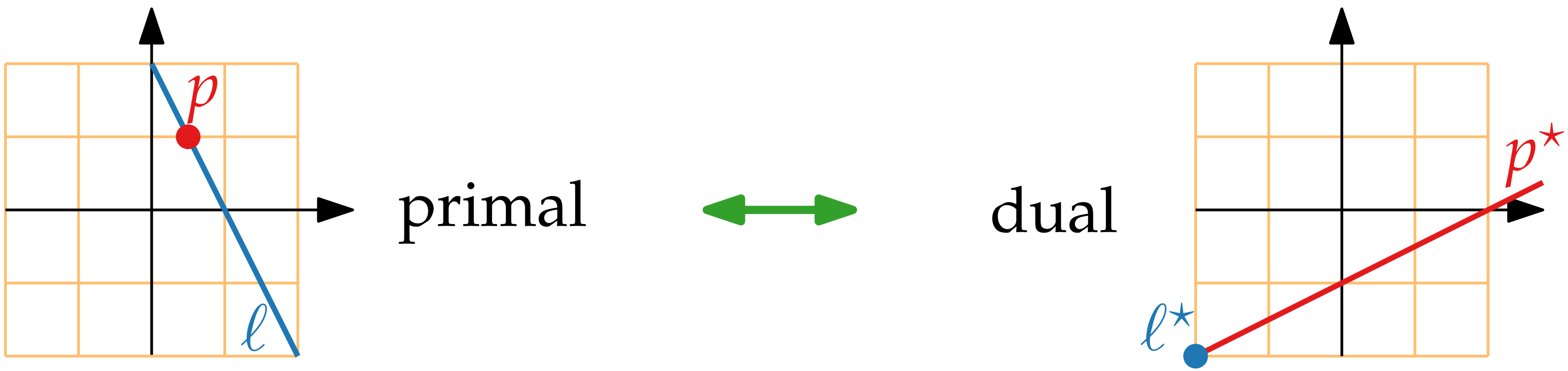
For $\ell\colon y = mx + b$, define $\ell^\star$ to be the pt q with $q^\star = \ell$,

that is, $\ell^\star = (m, -b)$.

Convex Hulls and Half-~~Space~~ Plane Intersections

Define dualtity $\star$ between pts and (non-vertical) lines:

For $p = (p_x, p_y)$, define the line $p^\star\colon y = p_x x - p_y$.



For $\ell\colon y = mx + b$, define $\ell^\star$ to be the pt q with $q^\star = \ell$,

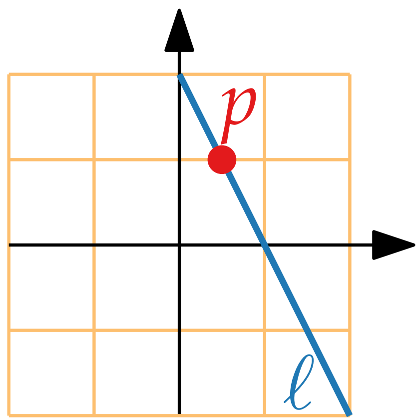
that is, $\ell^\star = (m, -b)$.

Observe.

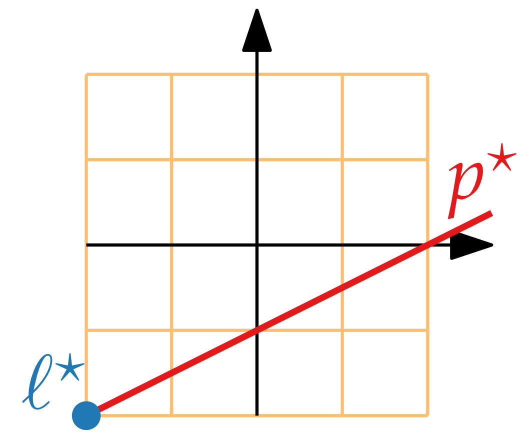
Convex Hulls and Half-~~Space~~ Plane Intersections

Define dualtity $\star$ between pts and (non-vertical) lines:

For $p = (p_x, p_y)$, define the line $p^\star\colon y = p_x x - p_y$.



primal $\longleftrightarrow$ dual



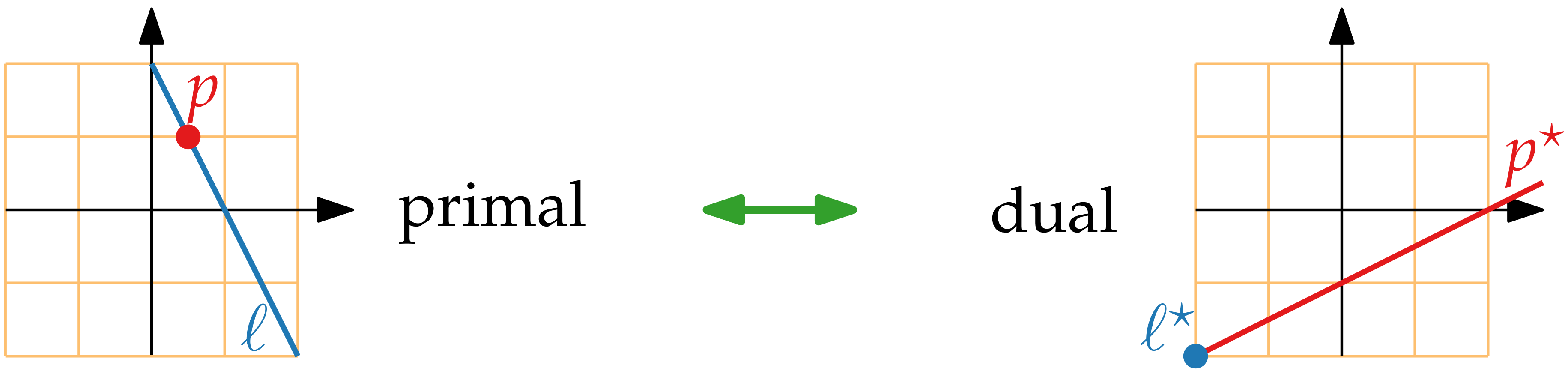
For $\ell\colon y = mx + b$, define $\ell^\star$ to be the pt q with $q^\star = \ell$, that is, $\ell^\star = (m, -b)$.

Observe. Let $p \in \mathbb{R}^2$ and let ℓ be a non-vertical line.

Convex Hulls and Half-~~Space~~ Plane Intersections

Define dualtity $\star$ between pts and (non-vertical) lines:

For $p = (p_x, p_y)$, define the line $p^\star \colon y = p_x x - p_y$.



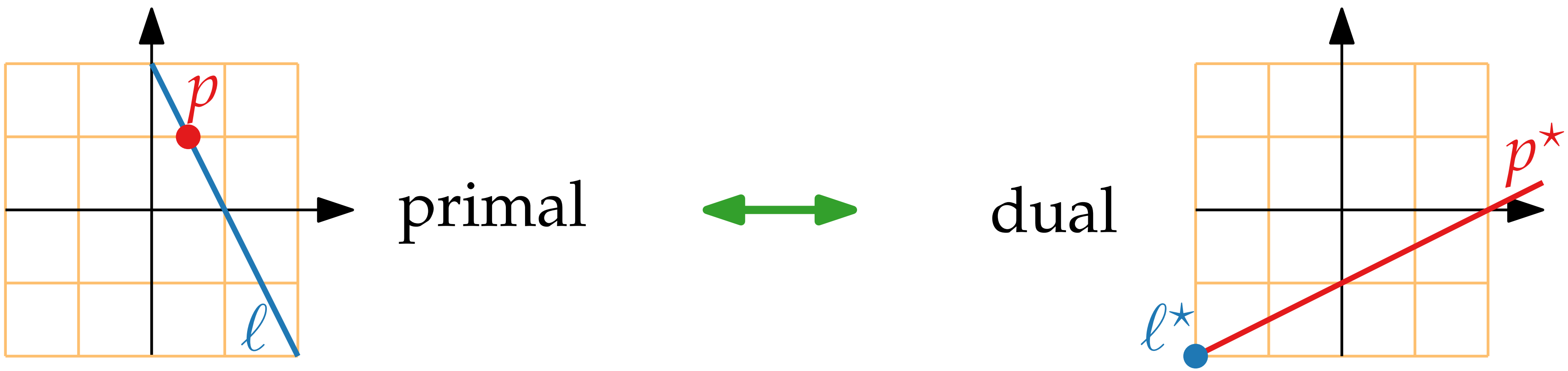
For $\ell \colon y = mx + b$, define $\ell^\star$ to be the pt q with $q^\star = \ell$,
that is, $\ell^\star = (m, -b)$.

Observe. Let $p \in \mathbb{R}^2$ and let ℓ be a non-vertical line.
$\star$ is incidence-preserving:

Convex Hulls and Half-~~Space~~ Plane Intersections

Define dualtity $\star$ between pts and (non-vertical) lines:

For $p = (p_x, p_y)$, define the line $p^\star \colon y = p_x x - p_y$.



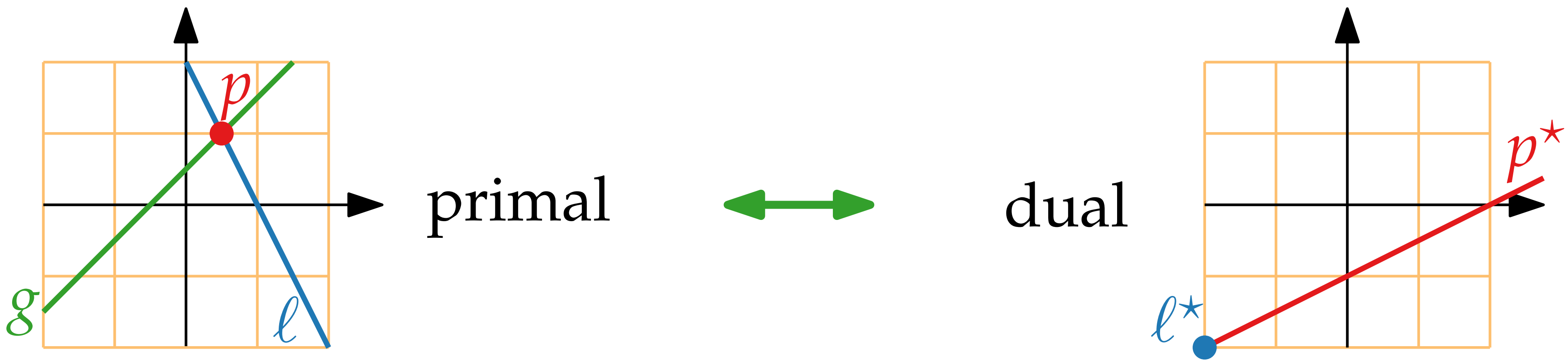
For $\ell \colon y = mx + b$, define $\ell^\star$ to be the pt q with $q^\star = \ell$,
that is, $\ell^\star = (m, -b)$.

Observe. Let $p \in \mathbb{R}^2$ and let ℓ be a non-vertical line.
$\star$ is incidence-preserving: $p \in \ell \;\Leftrightarrow\; \ell^\star \in p^\star$

Convex Hulls and Half-~~Space~~ Plane Intersections

Define dualtity $\star$ between pts and (non-vertical) lines:

For $p = (p_x, p_y)$, define the line $p^\star\colon y = p_x x - p_y$.



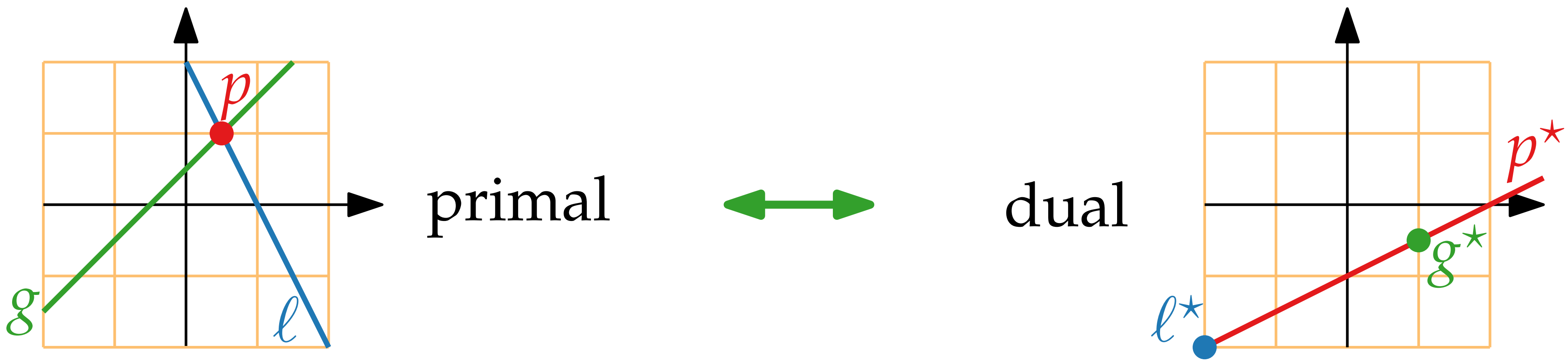
For $\ell\colon y = mx + b$, define $\ell^\star$ to be the pt q with $q^\star = \ell$,
that is, $\ell^\star = (m, -b)$.

Observe. Let $p \in \mathbb{R}^2$ and let ℓ be a non-vertical line.
$\star$ is incidence-preserving: $p \in \ell \;\Leftrightarrow\; \ell^\star \in p^\star$

Convex Hulls and Half-~~Space~~ Plane Intersections

Define dualtity $\star$ between pts and (non-vertical) lines:

For $p = (p_x, p_y)$, define the line $p^\star \colon y = p_x x - p_y$.



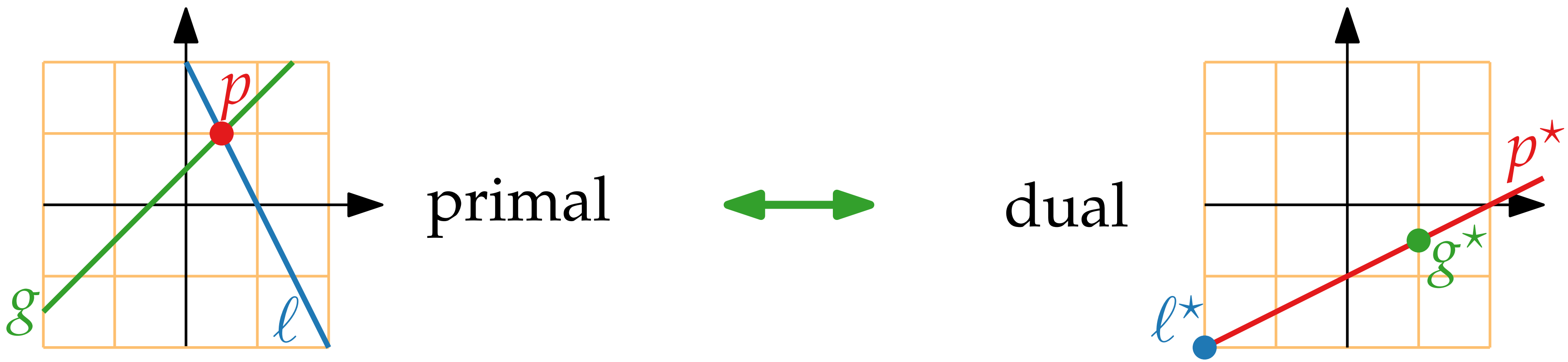
For $\ell \colon y = mx + b$, define $\ell^\star$ to be the pt q with $q^\star = \ell$,
that is, $\ell^\star = (m, -b)$.

Observe. Let $p \in \mathbb{R}^2$ and let ℓ be a non-vertical line.
$\star$ is incidence-preserving: $p \in \ell \;\Leftrightarrow\; \ell^\star \in p^\star$

Convex Hulls and Half-~~Space~~ Plane Intersections

Define dualtity $\star$ between pts and (non-vertical) lines:

For $p = (p_x, p_y)$, define the line $p^\star \colon y = p_x x - p_y$.



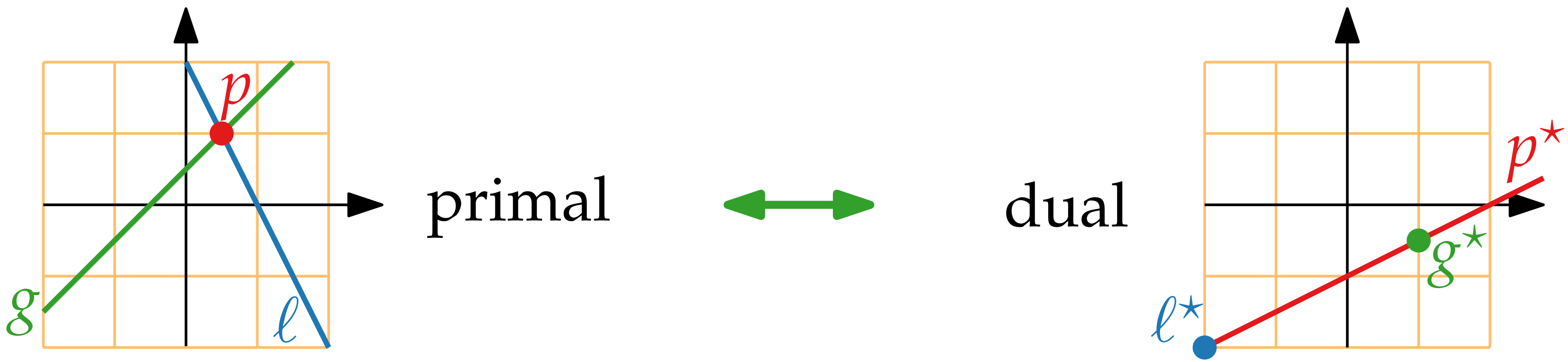
For $\ell \colon y = mx + b$, define $\ell^\star$ to be the pt q with $q^\star = \ell$,
that is, $\ell^\star = (m, -b)$.

Observe. Let $p \in \mathbb{R}^2$ and let ℓ be a non-vertical line.
$\star$ is incidence-preserving: $p \in \ell \;\Leftrightarrow\; \ell^\star \in p^\star$
$\star$ is order-preserving:

Convex Hulls and Half-~~Space~~ Plane Intersections

Define dualtity $\star$ between pts and (non-vertical) lines:

For $p = (p_x, p_y)$, define the line $p^\star \colon y = p_x x - p_y$.



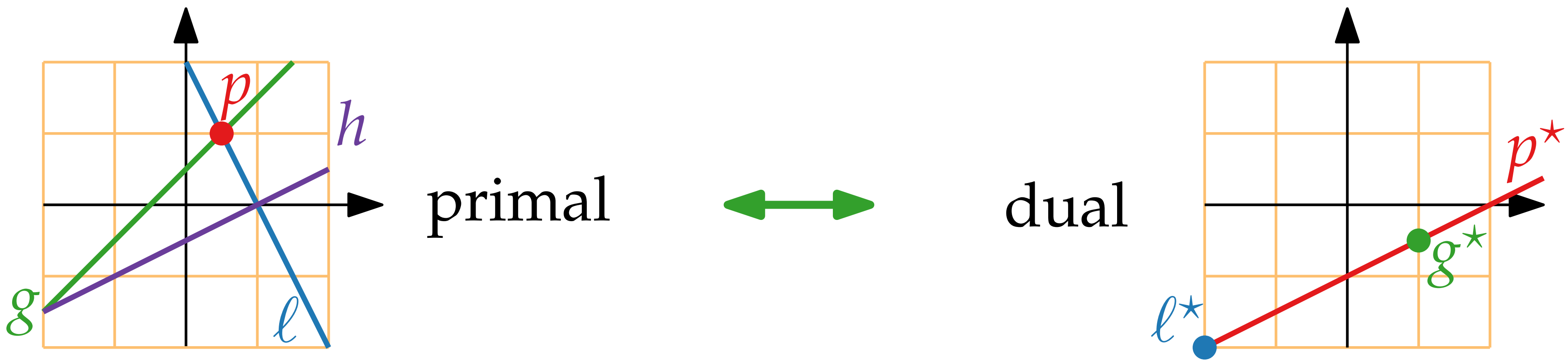
For $\ell \colon y = mx + b$, define $\ell^\star$ to be the pt q with $q^\star = \ell$,
that is, $\ell^\star = (m, -b)$.

Observe. Let $p \in \mathbb{R}^2$ and let ℓ be a non-vertical line.
$\star$ is incidence-preserving: $p \in \ell \;\Leftrightarrow\; \ell^\star \in p^\star$
$\star$ is order-preserving: p above $\ell \;\Leftrightarrow\; \ell^\star$ above $p^\star$

Convex Hulls and Half-~~Space~~ Plane Intersections

Define dualtity $\star$ between pts and (non-vertical) lines:

For $p = (p_x, p_y)$, define the line $p^\star\colon y = p_x x - p_y$.



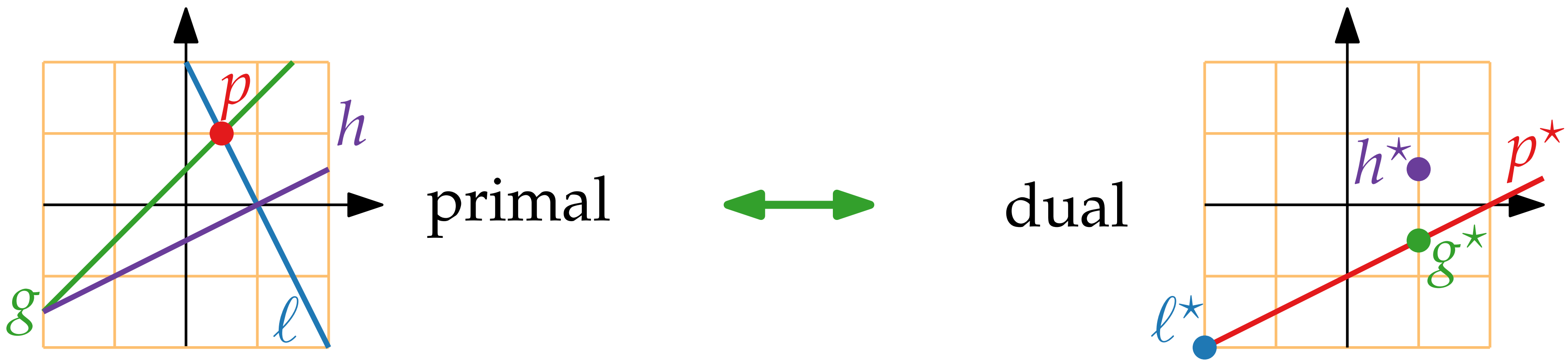
For $\ell\colon y = mx + b$, define $\ell^\star$ to be the pt q with $q^\star = \ell$,
that is, $\ell^\star = (m, -b)$.

Observe. Let $p \in \mathbb{R}^2$ and let ℓ be a non-vertical line.
$\star$ is incidence-preserving: $p \in \ell \;\Leftrightarrow\; \ell^\star \in p^\star$
$\star$ is order-preserving: p above $\ell \;\Leftrightarrow\; \ell^\star$ above $p^\star$

Convex Hulls and Half-~~Space~~ Plane Intersections

Define dualtity $\star$ between pts and (non-vertical) lines:

For $p = (p_x, p_y)$, define the line $p^\star \colon y = p_x x - p_y$.



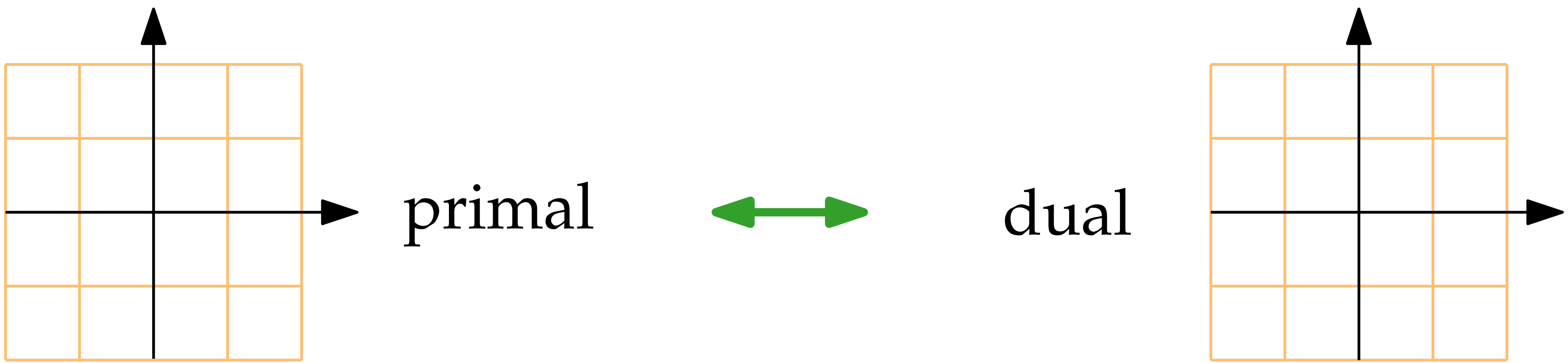
For $\ell \colon y = mx + b$, define $\ell^\star$ to be the pt q with $q^\star = \ell$,
that is, $\ell^\star = (m, -b)$.

Observe. Let $p \in \mathbb{R}^2$ and let ℓ be a non-vertical line.
$\star$ is incidence-preserving: $p \in \ell \iff \ell^\star \in p^\star$
$\star$ is order-preserving: p above $\ell \iff \ell^\star$ above $p^\star$

Convex Hulls and Half-~~Space~~ Plane Intersections

Define dualtity $\star$ between pts and (non-vertical) lines:

For $p = (p_x, p_y)$, define the line $p^\star\colon y = p_x x - p_y$.



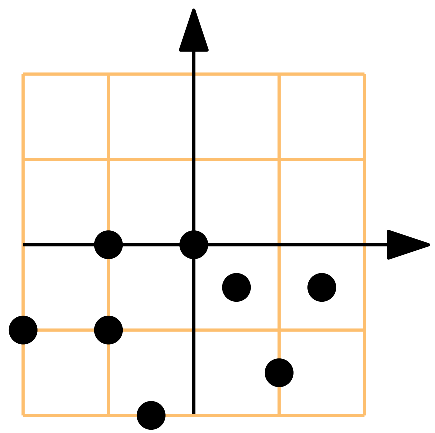
For $\ell\colon y = mx + b$, define $\ell^\star$ to be the pt q with $q^\star = \ell$,
that is, $\ell^\star = (m, -b)$.

Observe. Let $p \in \mathbb{R}^2$ and let ℓ be a non-vertical line.
$\star$ is incidence-preserving: $p \in \ell \iff \ell^\star \in p^\star$
$\star$ is order-preserving: p above $\ell \iff \ell^\star$ above $p^\star$

Convex Hulls and Half-~~Space~~ Plane Intersections

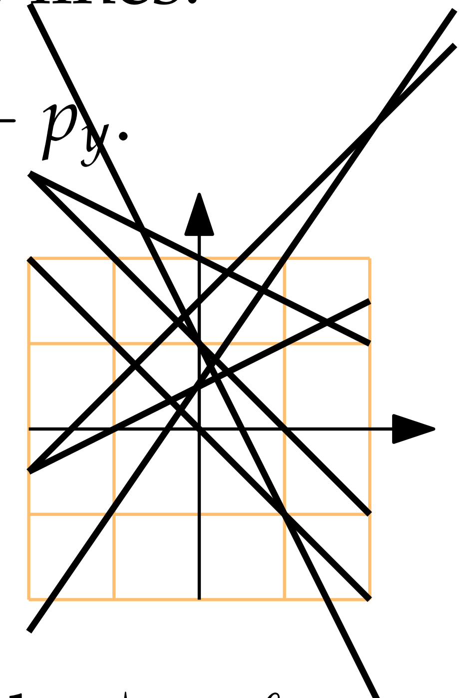
Define dualtity $\star$ between pts and (non-vertical) lines:

For $p = (p_x, p_y)$, define the line $p^\star \colon y = p_x x - p_y$.



primal

dual



For $\ell \colon y = mx + b$, define $\ell^\star$ to be the pt q with $q^\star = \ell$,

that is, $\ell^\star = (m, -b)$.

Observe. Let $p \in \mathbb{R}^2$ and let ℓ be a non-vertical line.

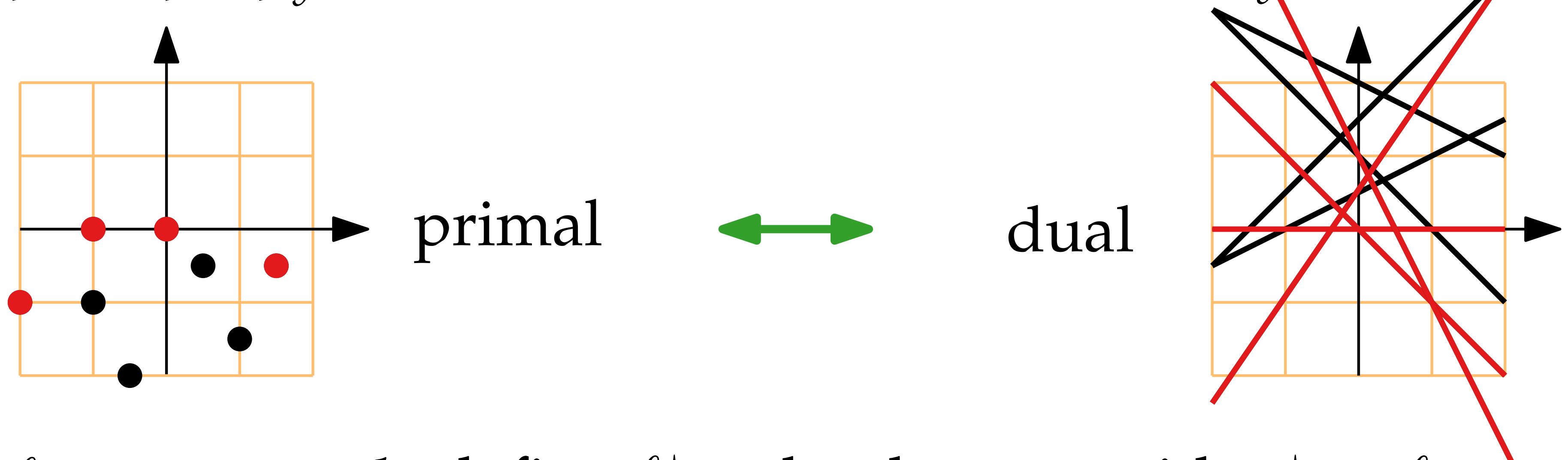
$\star$ is incidence-preserving: $p \in \ell \;\Leftrightarrow\; \ell^\star \in p^\star$

$\star$ is order-preserving: p above $\ell \;\Leftrightarrow\; \ell^\star$ above $p^\star$

Convex Hulls and Half-~~Space~~ Plane Intersections

Define dualtity $\star$ between pts and (non-vertical) lines:

For $p = (p_x, p_y)$, define the line $p^\star \colon y = p_x x - p_y$.



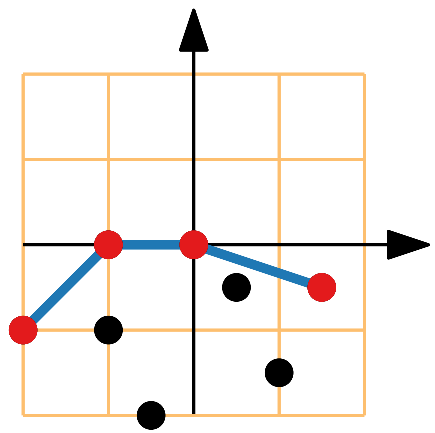
For $\ell \colon y = mx + b$, define $\ell^\star$ to be the pt q with $q^\star = \ell$,
that is, $\ell^\star = (m, -b)$.

Observe. Let $p \in \mathbb{R}^2$ and let ℓ be a non-vertical line.
$\star$ is incidence-preserving: $p \in \ell \iff \ell^\star \in p^\star$
$\star$ is order-preserving: p above $\ell \iff \ell^\star$ above $p^\star$

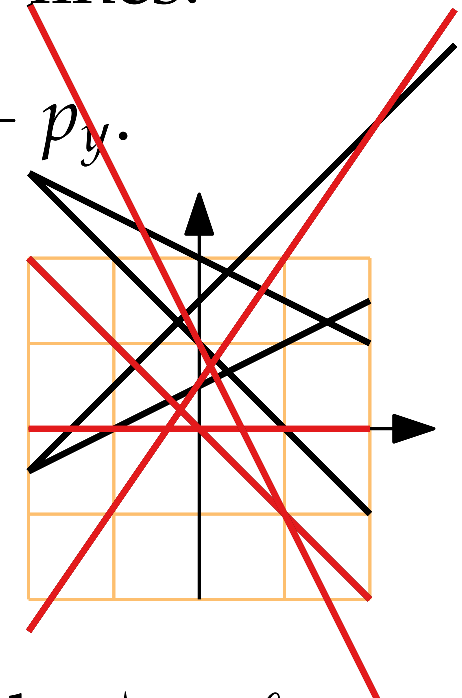
Convex Hulls and Half-~~Space~~ Plane Intersections

Define dualtity $\star$ between pts and (non-vertical) lines:

For $p = (p_x, p_y)$, define the line $p^\star \colon y = p_x x - p_y$.



primal $\longleftrightarrow$ dual



For $\ell \colon y = mx + b$, define $\ell^\star$ to be the pt q with $q^\star = \ell$,

that is, $\ell^\star = (m, -b)$.

Observe. Let $p \in \mathbb{R}^2$ and let ℓ be a non-vertical line.

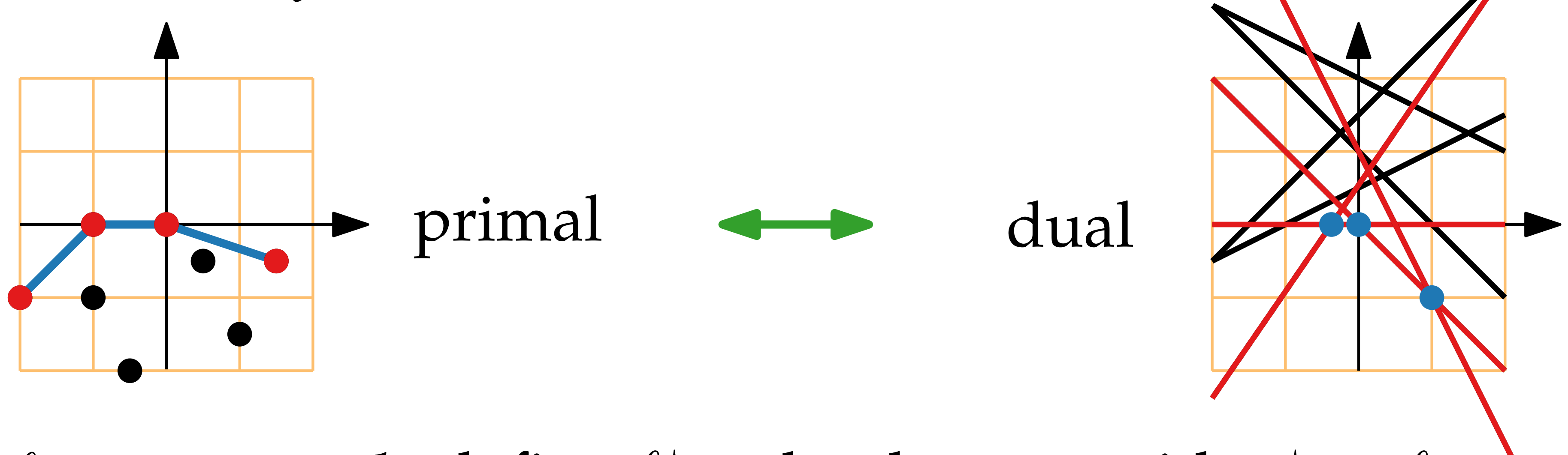
$\star$ is incidence-preserving: $p \in \ell \;\Leftrightarrow\; \ell^\star \in p^\star$

$\star$ is order-preserving: p above $\ell \;\Leftrightarrow\; \ell^\star$ above $p^\star$

Convex Hulls and Half-~~Space~~ Plane Intersections

Define dualtity $\star$ between pts and (non-vertical) lines:

For $p = (p_x, p_y)$, define the line $p^\star \colon y = p_x x - p_y$.



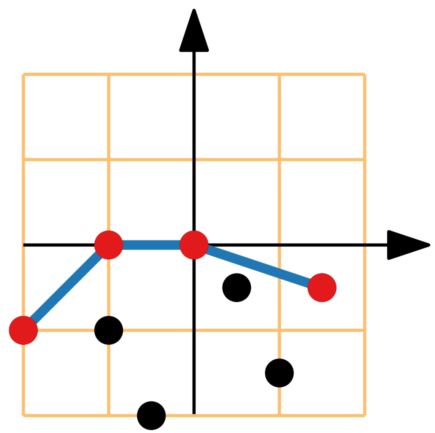
For $\ell \colon y = mx + b$, define $\ell^\star$ to be the pt q with $q^\star = \ell$,
that is, $\ell^\star = (m, -b)$.

Observe. Let $p \in \mathbb{R}^2$ and let ℓ be a non-vertical line.
$\star$ is incidence-preserving: $p \in \ell \;\Leftrightarrow\; \ell^\star \in p^\star$
$\star$ is order-preserving: p above $\ell \;\Leftrightarrow\; \ell^\star$ above $p^\star$

Convex Hulls and Half-~~Space~~ Plane Intersections

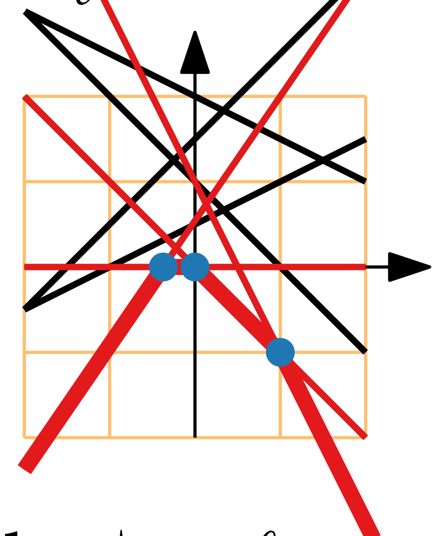
Define dualtity $\star$ between pts and (non-vertical) lines:

For $p = (p_x, p_y)$, define the line $p^\star \colon y = p_x x - p_y$.



primal

dual



For $\ell \colon y = mx + b$, define $\ell^\star$ to be the pt q with $q^\star = \ell$,

that is, $\ell^\star = (m, -b)$.

Observe. Let $p \in \mathbb{R}^2$ and let ℓ be a non-vertical line.

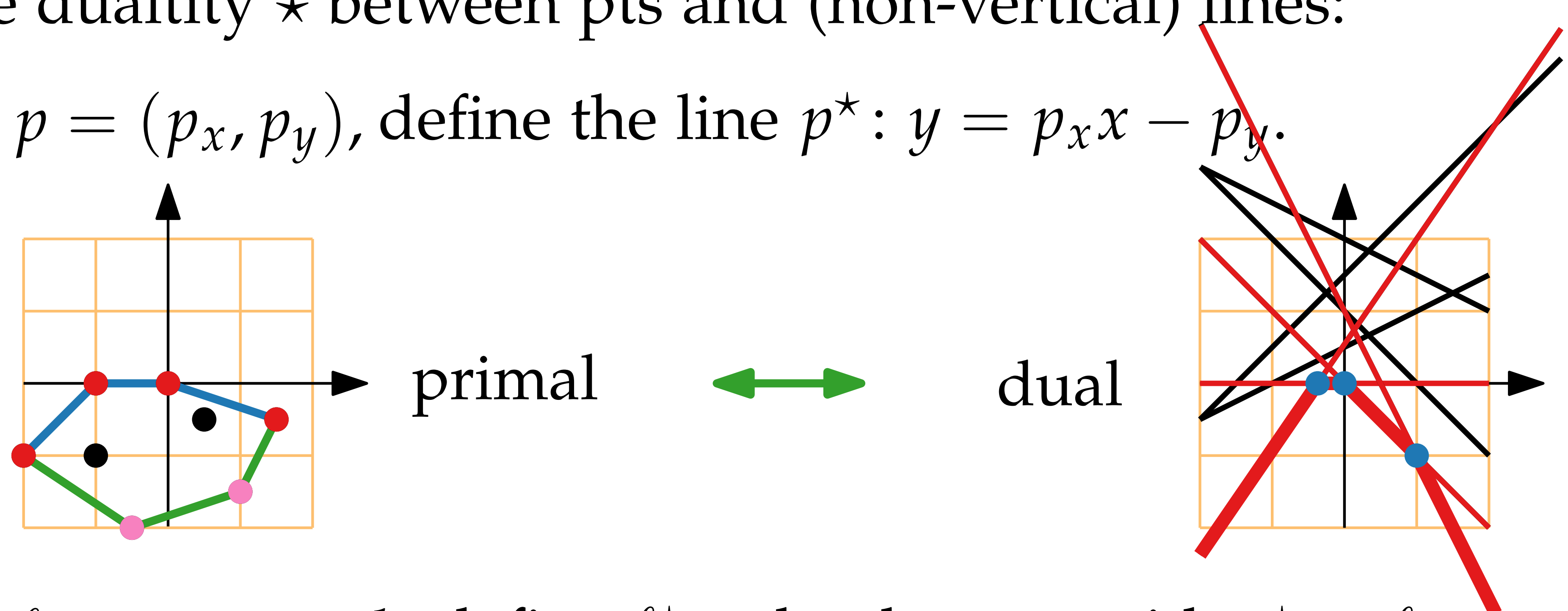
$\star$ is incidence-preserving: $p \in \ell \;\Leftrightarrow\; \ell^\star \in p^\star$

$\star$ is order-preserving: p above $\ell \;\Leftrightarrow\; \ell^\star$ above $p^\star$

Convex Hulls and Half-~~Space~~ Plane Intersections

Define dualtity $\star$ between pts and (non-vertical) lines:

For $p = (p_x, p_y)$, define the line $p^\star\colon y = p_x x - p_y$.



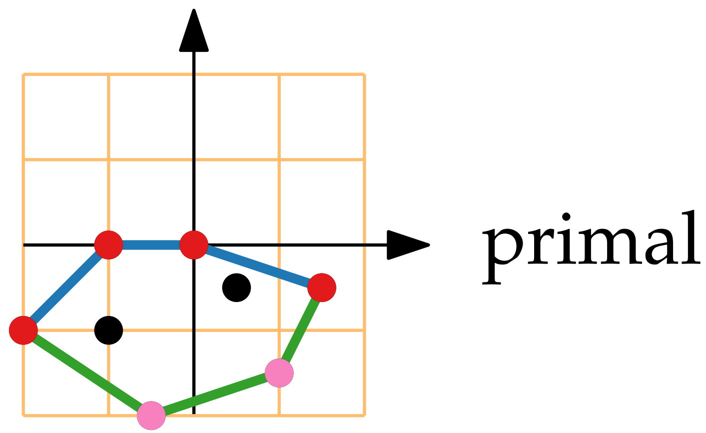
For $\ell\colon y = mx + b$, define $\ell^\star$ to be the pt q with $q^\star = \ell$,
that is, $\ell^\star = (m, -b)$.

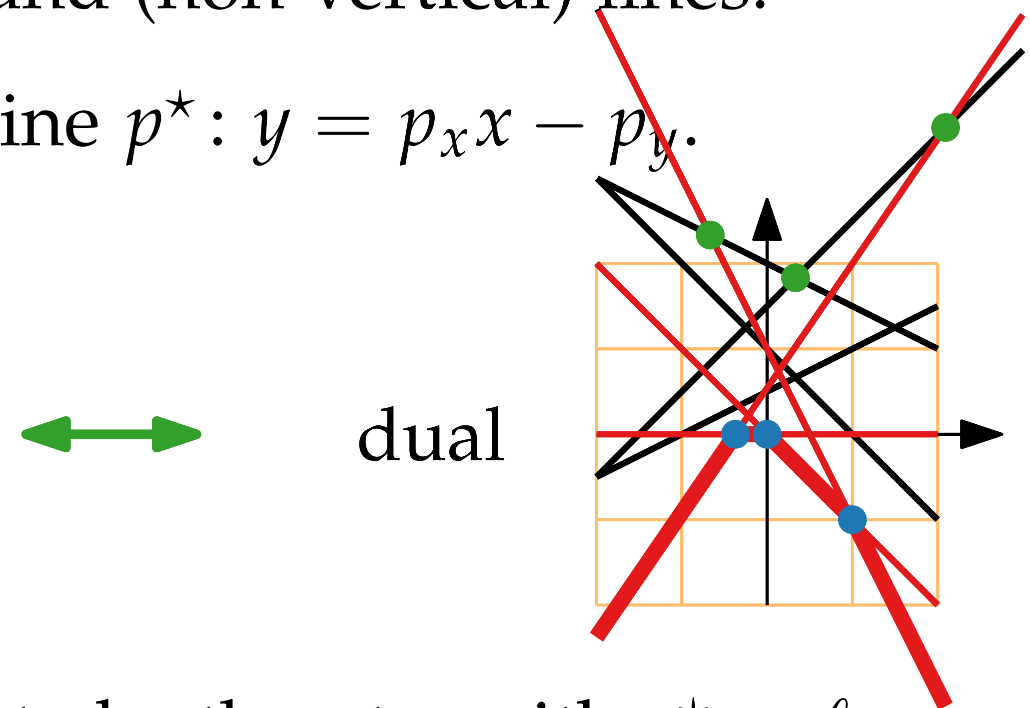
Observe. Let $p \in \mathbb{R}^2$ and let ℓ be a non-vertical line.
$\star$ is incidence-preserving: $p \in \ell \;\Leftrightarrow\; \ell^\star \in p^\star$
$\star$ is order-preserving: p above $\ell \;\Leftrightarrow\; \ell^\star$ above $p^\star$

Convex Hulls and Half-~~Space~~ Plane Intersections

Define dualtity $\star$ between pts and (non-vertical) lines:

For $p = (p_x, p_y)$, define the line $p^\star \colon y = p_x x - p_y$.





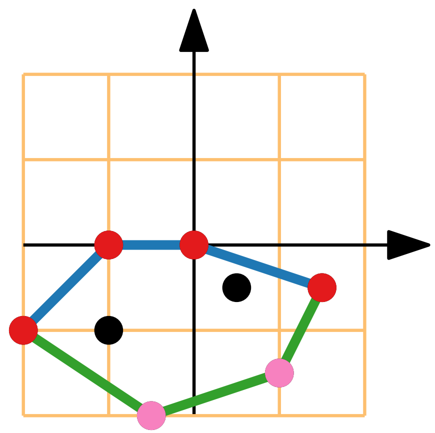
For $\ell \colon y = mx + b$, define $\ell^\star$ to be the pt q with $q^\star = \ell$,
that is, $\ell^\star = (m, -b)$.

Observe. Let $p \in \mathbb{R}^2$ and let ℓ be a non-vertical line.
$\star$ is incidence-preserving: $p \in \ell \;\Leftrightarrow\; \ell^\star \in p^\star$
$\star$ is order-preserving: p above $\ell \;\Leftrightarrow\; \ell^\star$ above $p^\star$

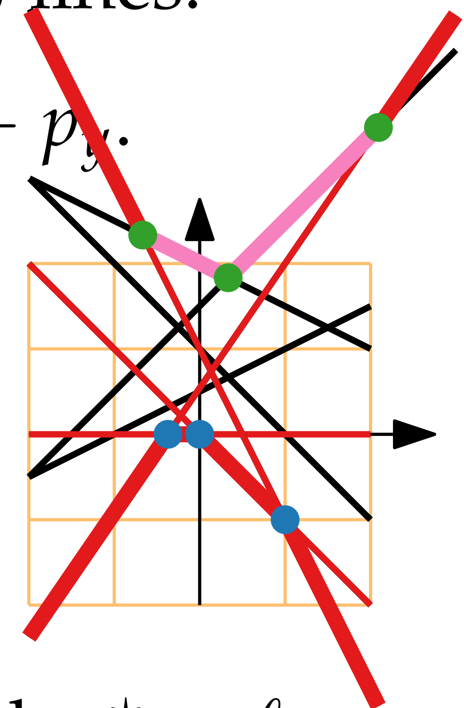
Convex Hulls and Half-~~Space~~ Plane Intersections

Define dualtity $\star$ between pts and (non-vertical) lines:

For $p = (p_x, p_y)$, define the line $p^\star \colon y = p_x x - p_y$.



primal $\longleftrightarrow$ dual



For $\ell \colon y = mx + b$, define $\ell^\star$ to be the pt q with $q^\star = \ell$, that is, $\ell^\star = (m, -b)$.

Observe. Let $p \in \mathbb{R}^2$ and let ℓ be a non-vertical line.

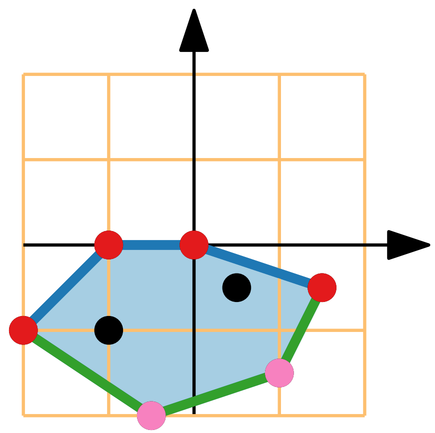
$\star$ is incidence-preserving: $p \in \ell \;\Leftrightarrow\; \ell^\star \in p^\star$

$\star$ is order-preserving: p above $\ell \;\Leftrightarrow\; \ell^\star$ above $p^\star$

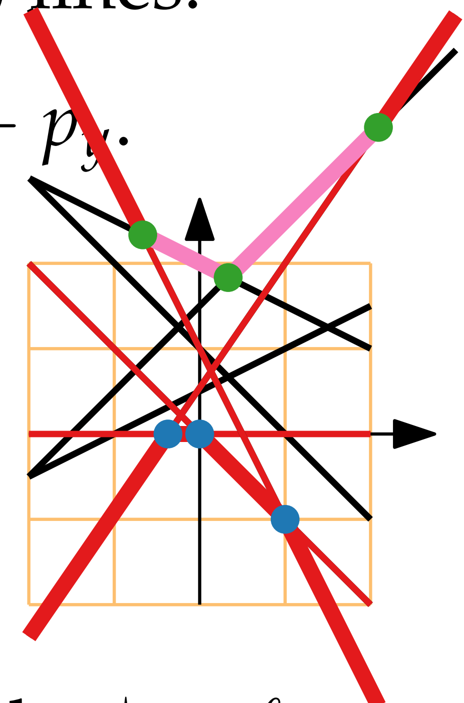
Convex Hulls and Half-~~Space~~ Plane Intersections

Define dualtity $\star$ between pts and (non-vertical) lines:

For $p = (p_x, p_y)$, define the line $p^\star \colon y = p_x x - p_y$.



primal $\longleftrightarrow$ dual



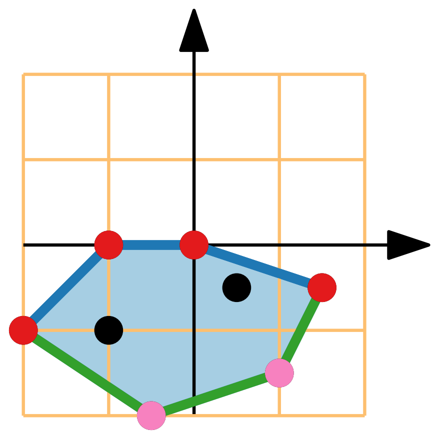
For $\ell \colon y = mx + b$, define $\ell^\star$ to be the pt q with $q^\star = \ell$, that is, $\ell^\star = (m, -b)$.

Observe. Let $p \in \mathbb{R}^2$ and let ℓ be a non-vertical line.
$\star$ is incidence-preserving: $p \in \ell \iff \ell^\star \in p^\star$
$\star$ is order-preserving: p above $\ell \iff \ell^\star$ above $p^\star$

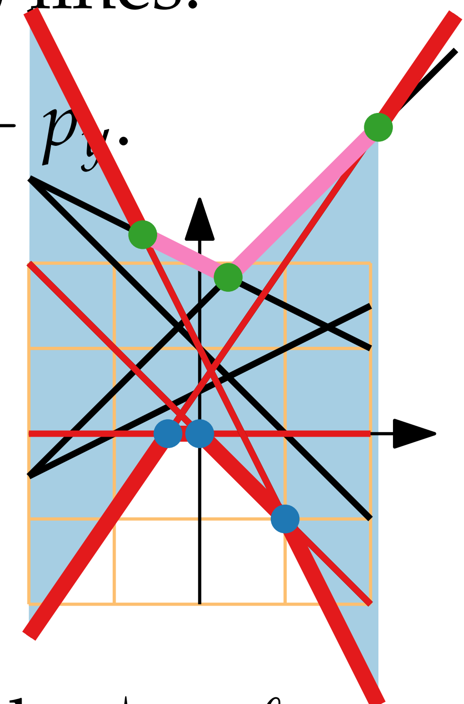
Convex Hulls and Half-~~Space~~ Plane Intersections

Define dualtity $\star$ between pts and (non-vertical) lines:

For $p = (p_x, p_y)$, define the line $p^\star \colon y = p_x x - p_y$.



primal $\longleftrightarrow$ dual



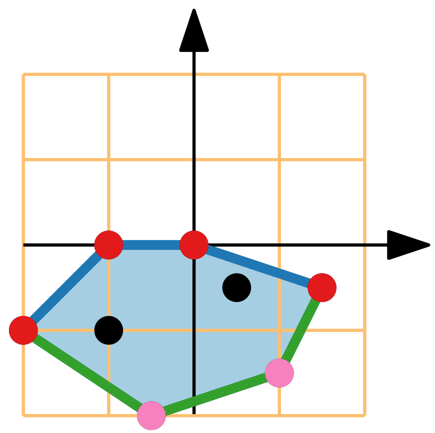
For $\ell \colon y = mx + b$, define $\ell^\star$ to be the pt q with $q^\star = \ell$,
that is, $\ell^\star = (m, -b)$.

Observe. Let $p \in \mathbb{R}^2$ and let ℓ be a non-vertical line.
$\star$ is incidence-preserving: $p \in \ell \;\Leftrightarrow\; \ell^\star \in p^\star$
$\star$ is order-preserving: p above $\ell \;\Leftrightarrow\; \ell^\star$ above $p^\star$

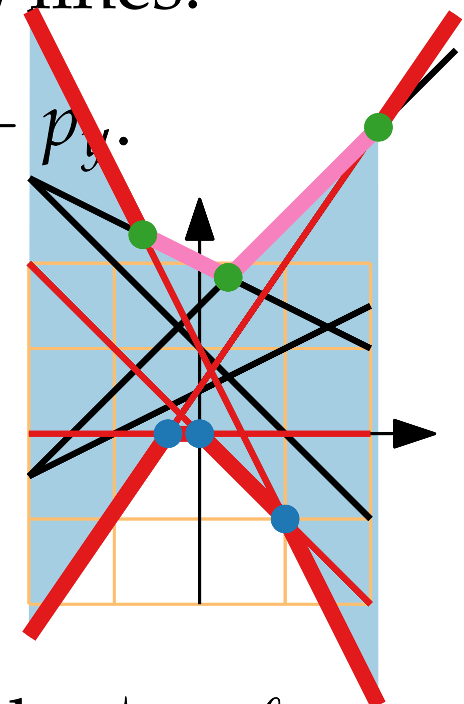
Convex Hulls and Half-~~Space~~ Plane Intersections

Define dualtity $\star$ between pts and (non-vertical) lines:

For $p = (p_x, p_y)$, define the line $p^\star\colon y = p_x x - p_y$.



primal $\longleftrightarrow$ dual



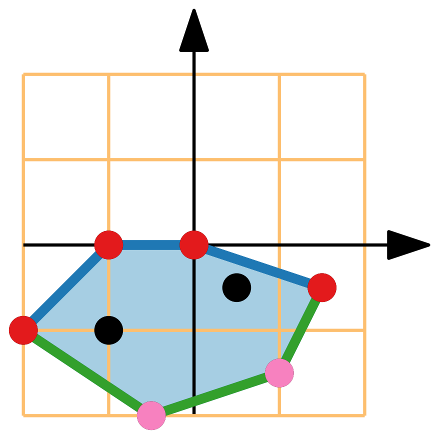
For $\ell\colon y = mx + b$, define $\ell^\star$ to be the pt q with $q^\star = \ell$,
that is, $\ell^\star = (m, -b)$.

Observe. ■ upper convex hulls of pts $\leftrightarrow$ lower env. of lines

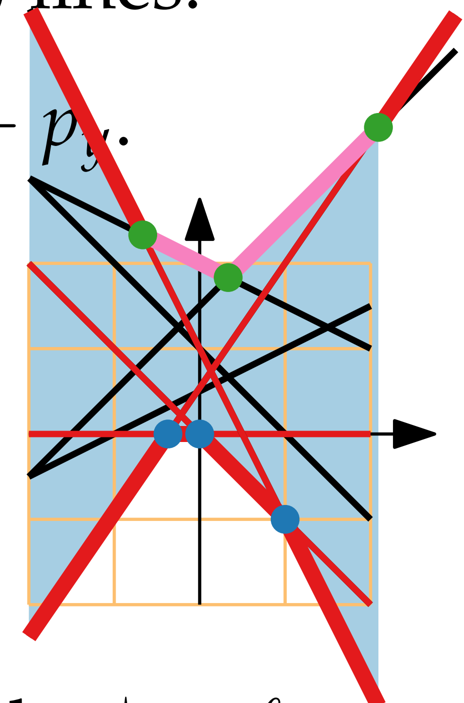
Convex Hulls and Half-~~Space~~ Plane Intersections

Define dualtity $\star$ between pts and (non-vertical) lines:

For $p = (p_x, p_y)$, define the line $p^\star \colon y = p_x x - p_y$.



$\longleftrightarrow$



For $\ell \colon y = mx + b$, define $\ell^\star$ to be the pt q with $q^\star = \ell$,
that is, $\ell^\star = (m, -b)$.

Observe.
- upper convex hulls of pts $\leftrightarrow$ lower env. of lines
- can compute inters. of "lower/upper" half planes (spaces) via upper/lower convex hulls

Computational Geometry

Lecture 9:
Convex Hulls in 3D
or
Mixing More Things

Part V:
Voronoi Diagrams Revisited

Philipp Kindermann

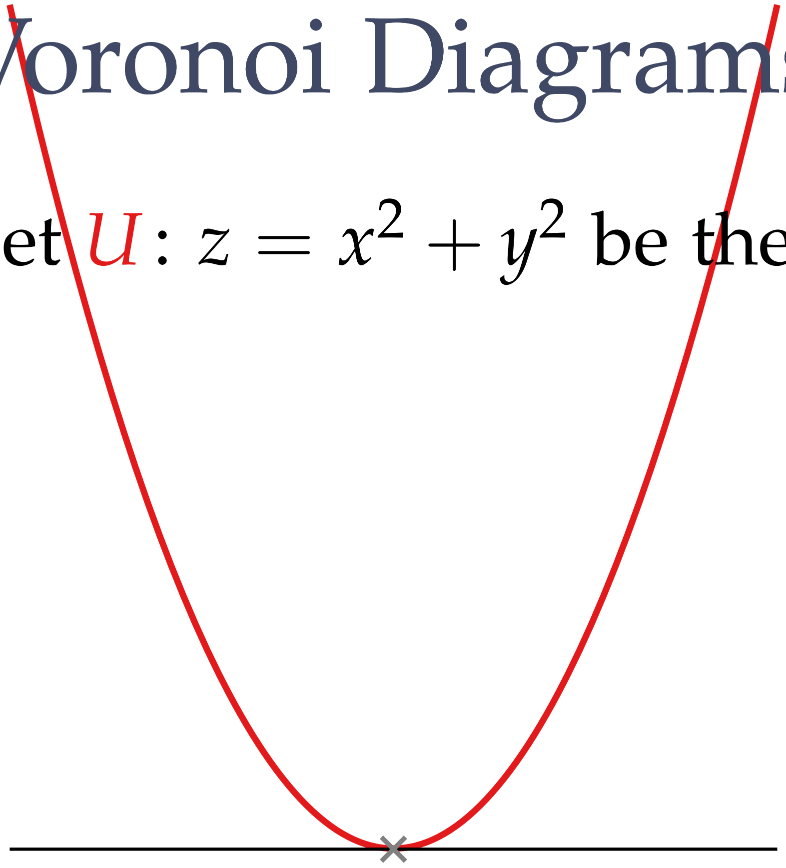
Winter Semester 2020

Voronoi Diagrams Revisited

Let $\color{red}{U}\colon z = x^2 + y^2$ be the *unit paraboloid* in $\mathbb{R}^3$.

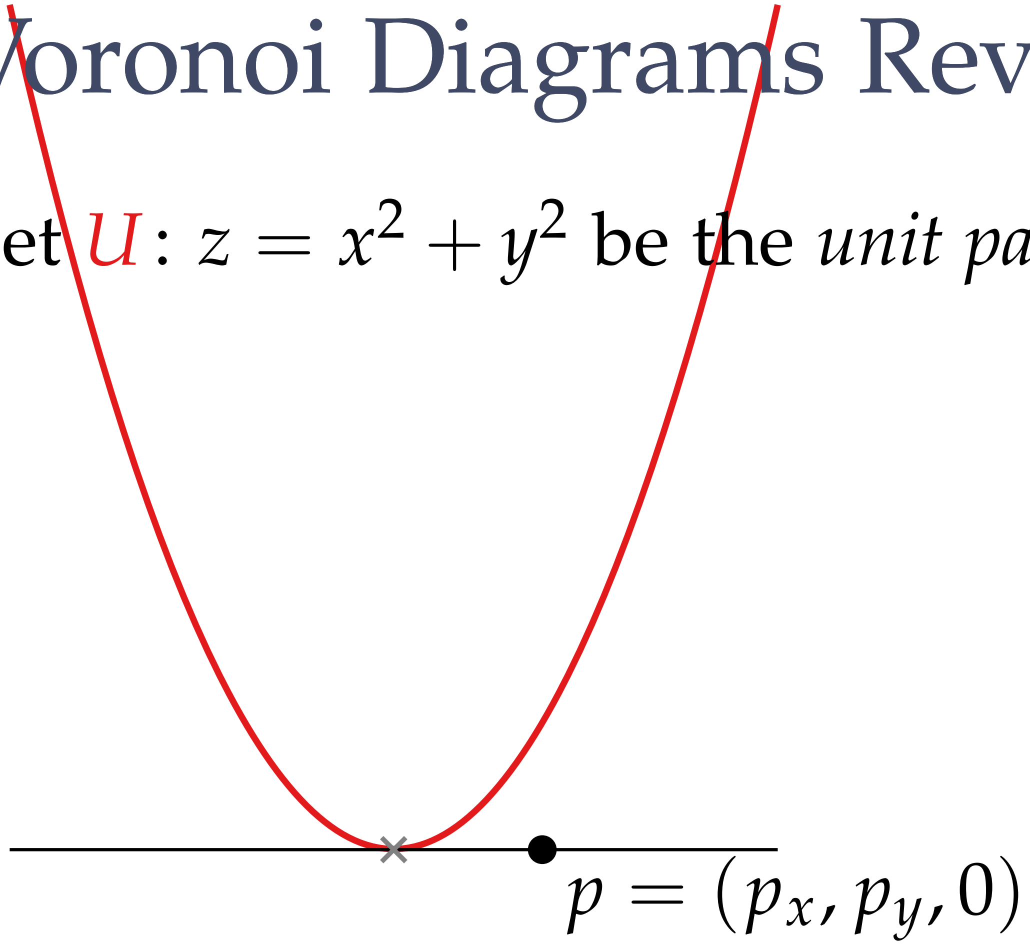
Voronoi Diagrams Revisited

Let U: $z = x^2 + y^2$ be the *unit paraboloid* in $\mathbb{R}^3$.



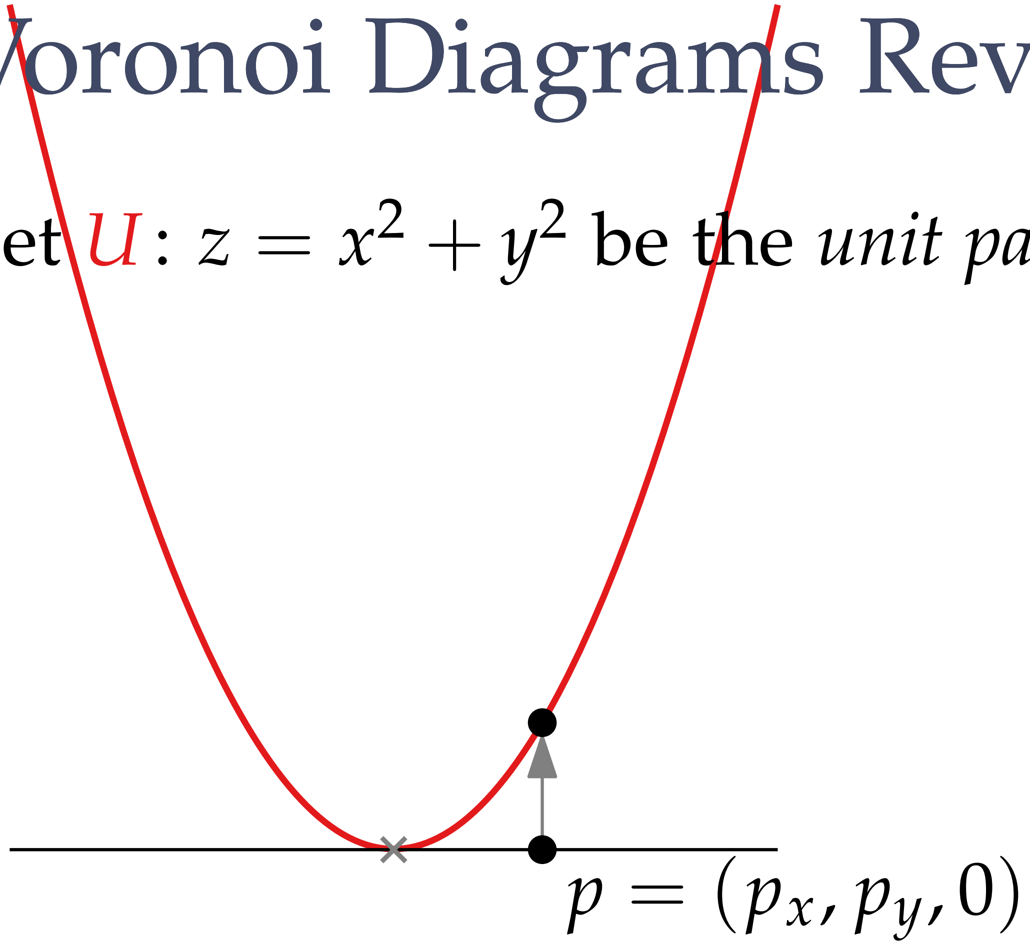
Voronoi Diagrams Revisited

Let $U\colon z = x^2 + y^2$ be the *unit paraboloid* in $\mathbb{R}^3$.



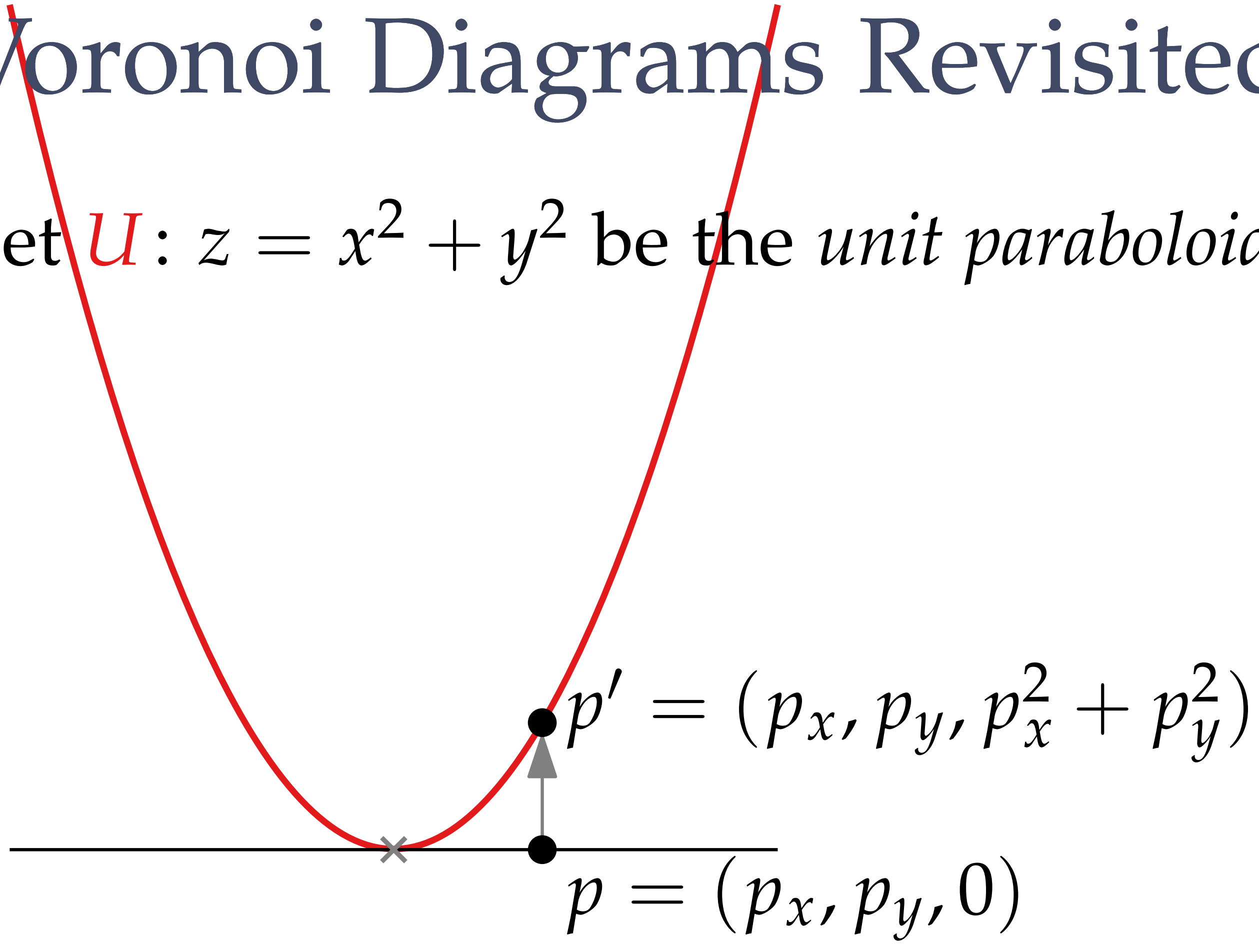
Voronoi Diagrams Revisited

Let $U: z = x^2 + y^2$ be the *unit paraboloid* in $\mathbb{R}^3$.



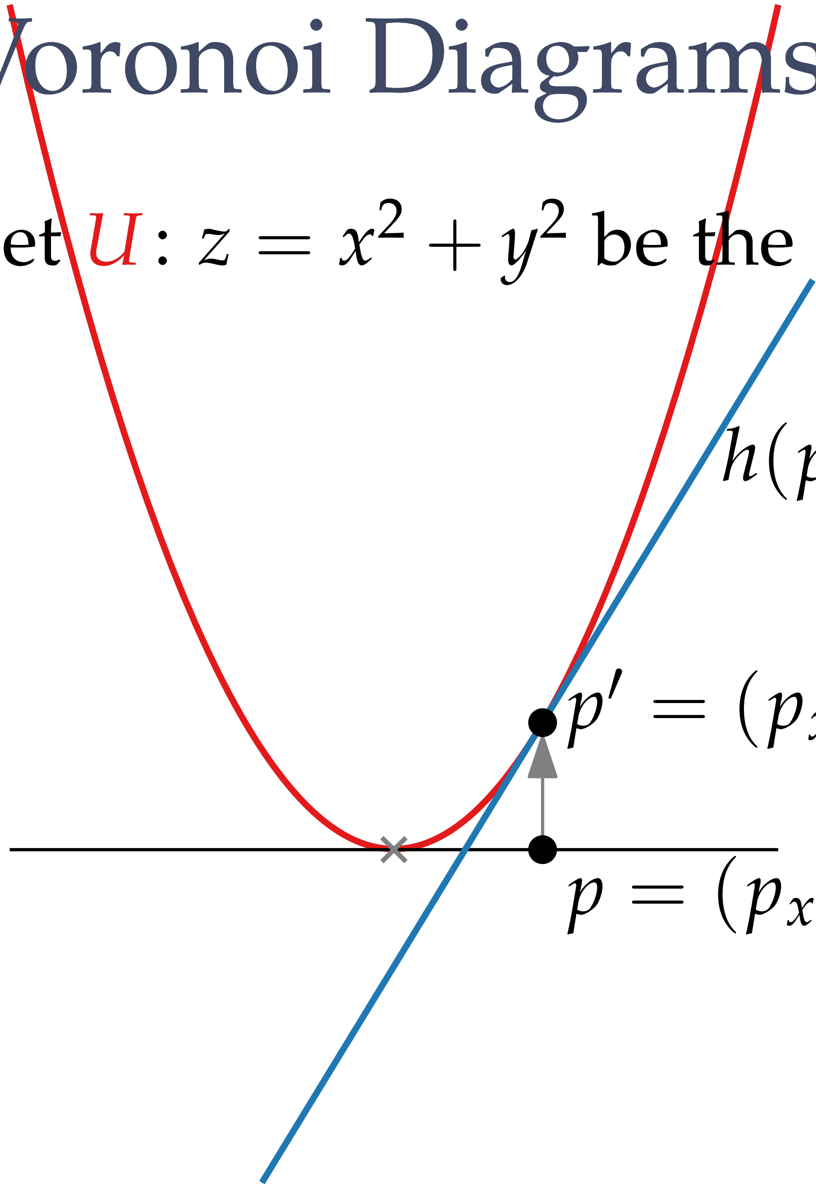
Voronoi Diagrams Revisited

Let $U\colon z = x^2 + y^2$ be the *unit paraboloid* in $\mathbb{R}^3$.



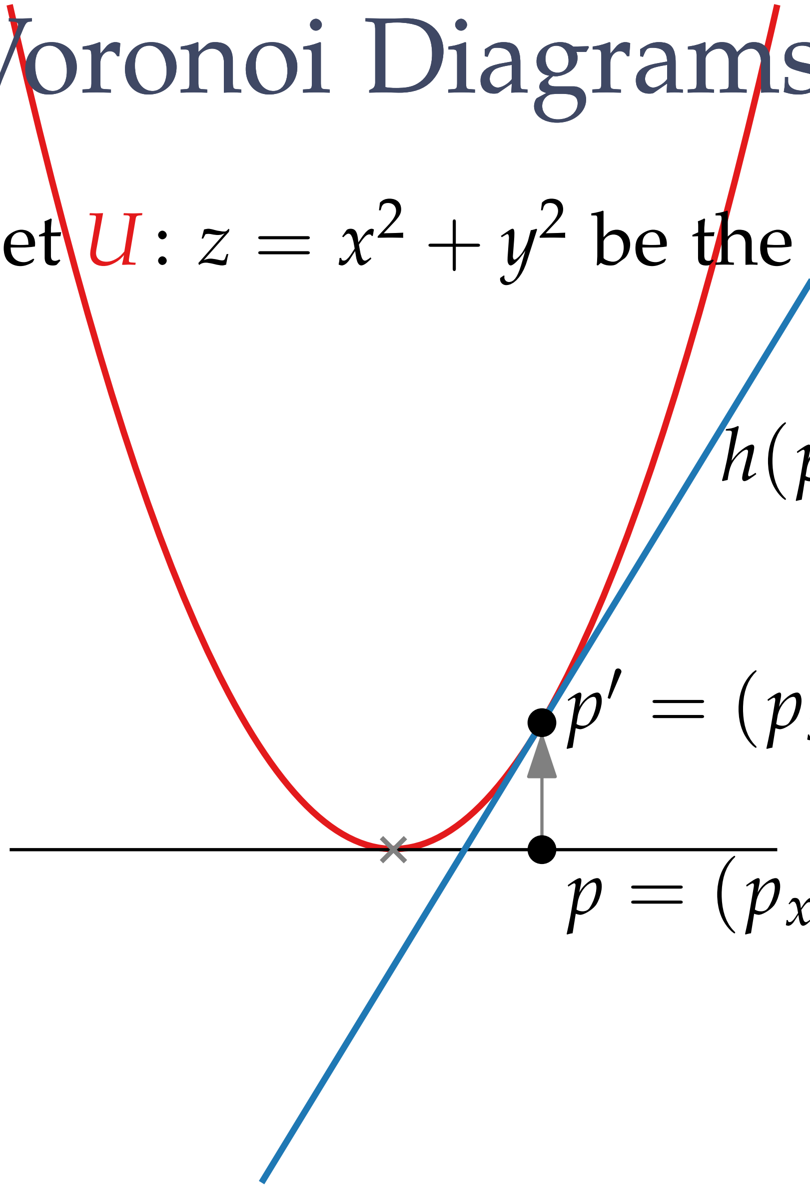
Voronoi Diagrams Revisited

Let $U\colon z = x^2 + y^2$ be the *unit paraboloid* in $\mathbb{R}^3$.



Voronoi Diagrams Revisited

Let $U\colon z = x^2 + y^2$ be the *unit paraboloid* in $\mathbb{R}^3$.

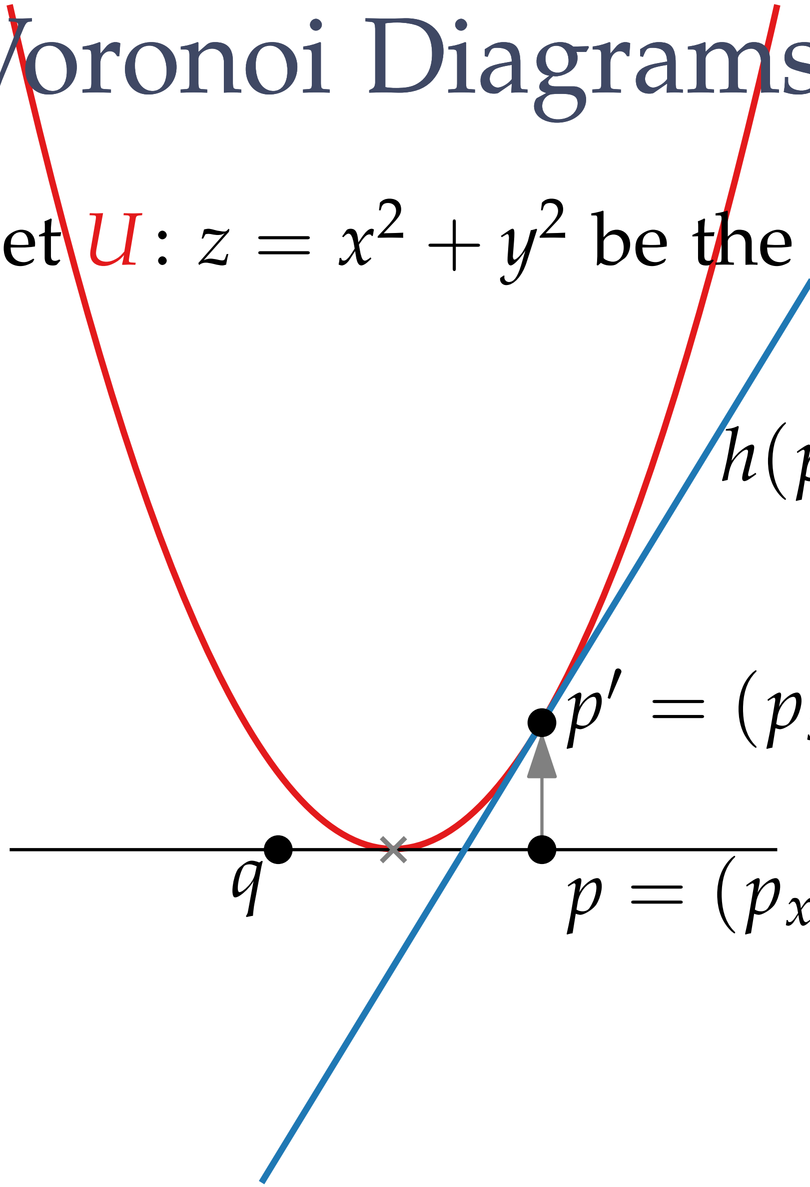


$h(p)\colon z = (2p_x)x + (2p_y)y - (p_x^2 + p_y^2)$

Note that $p' \in h(p)$.

Voronoi Diagrams Revisited

Let $\color{red}{U}\colon z = x^2 + y^2$ be the *unit paraboloid* in $\mathbb{R}^3$.



$$h(p)\colon z = (2p_x)x + (2p_y)y - (p_x^2 + p_y^2)$$

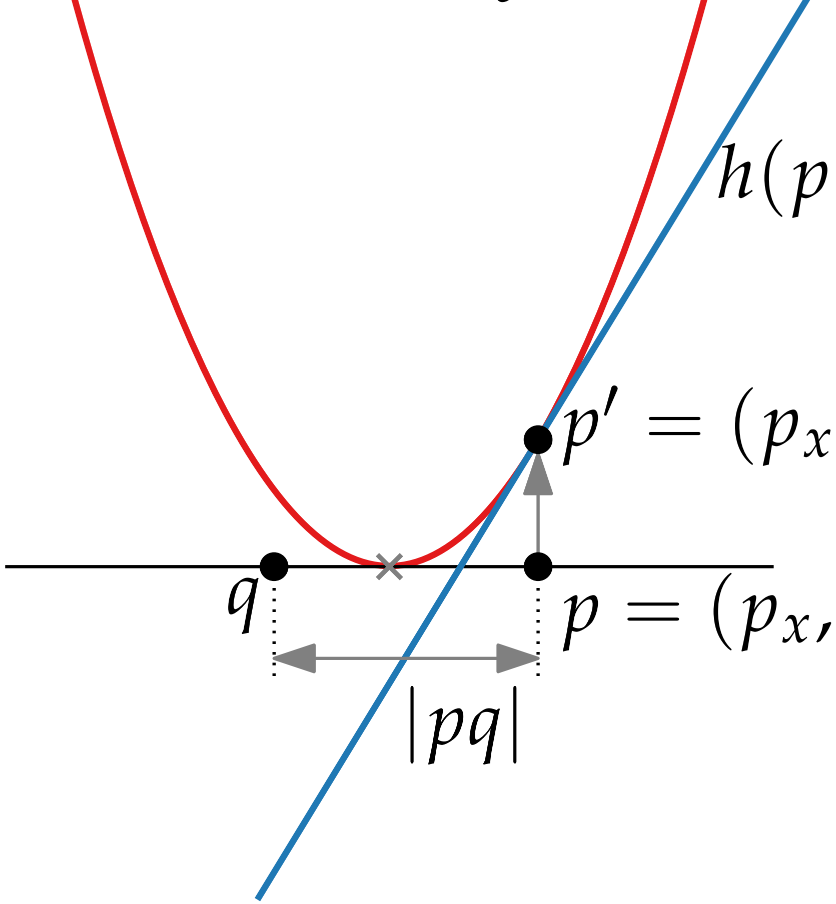
Note that $p' \in h(p)$.

Voronoi Diagrams Revisited

Let $U: z = x^2 + y^2$ be the *unit paraboloid* in $\mathbb{R}^3$.

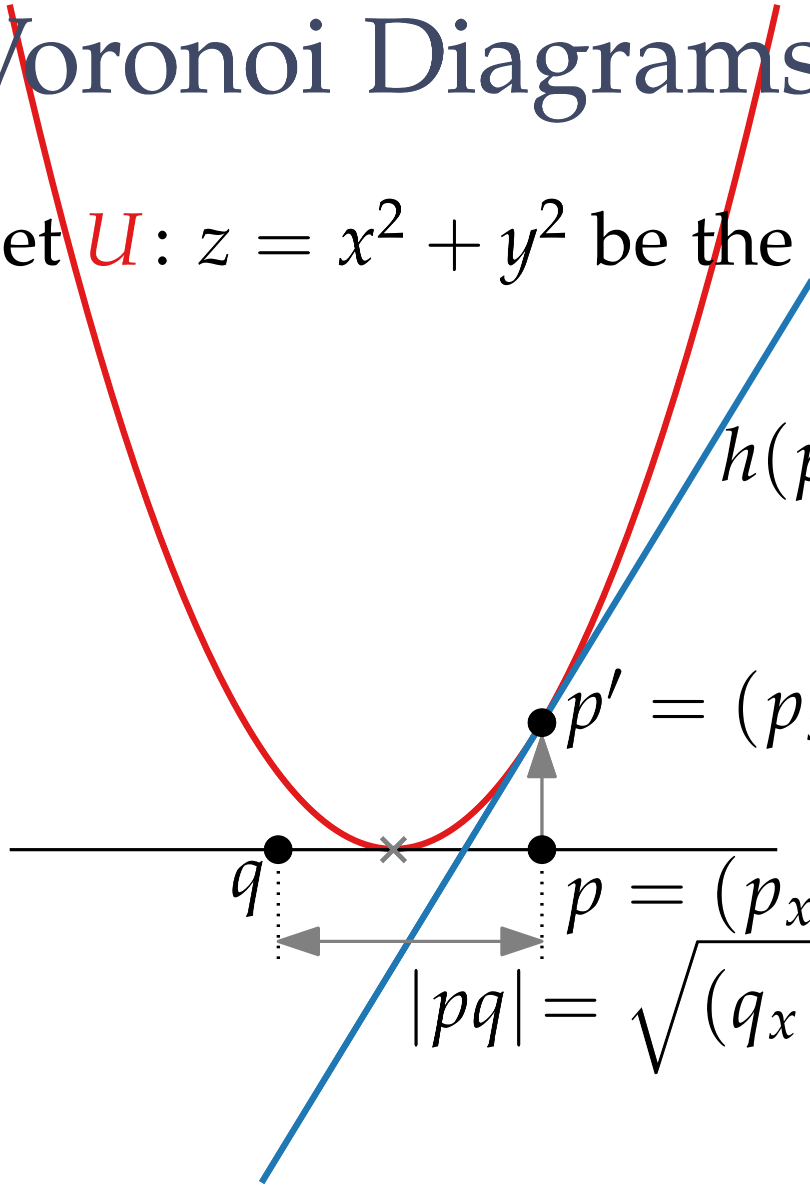
$$h(p): z = (2p_x)x + (2p_y)y - (p_x^2 + p_y^2)$$

Note that $p' \in h(p)$.



Voronoi Diagrams Revisited

Let $U : z = x^2 + y^2$ be the *unit paraboloid* in $\mathbb{R}^3$.



$h(p) : z = (2p_x)x + (2p_y)y - (p_x^2 + p_y^2)$

Note that $p' \in h(p)$.

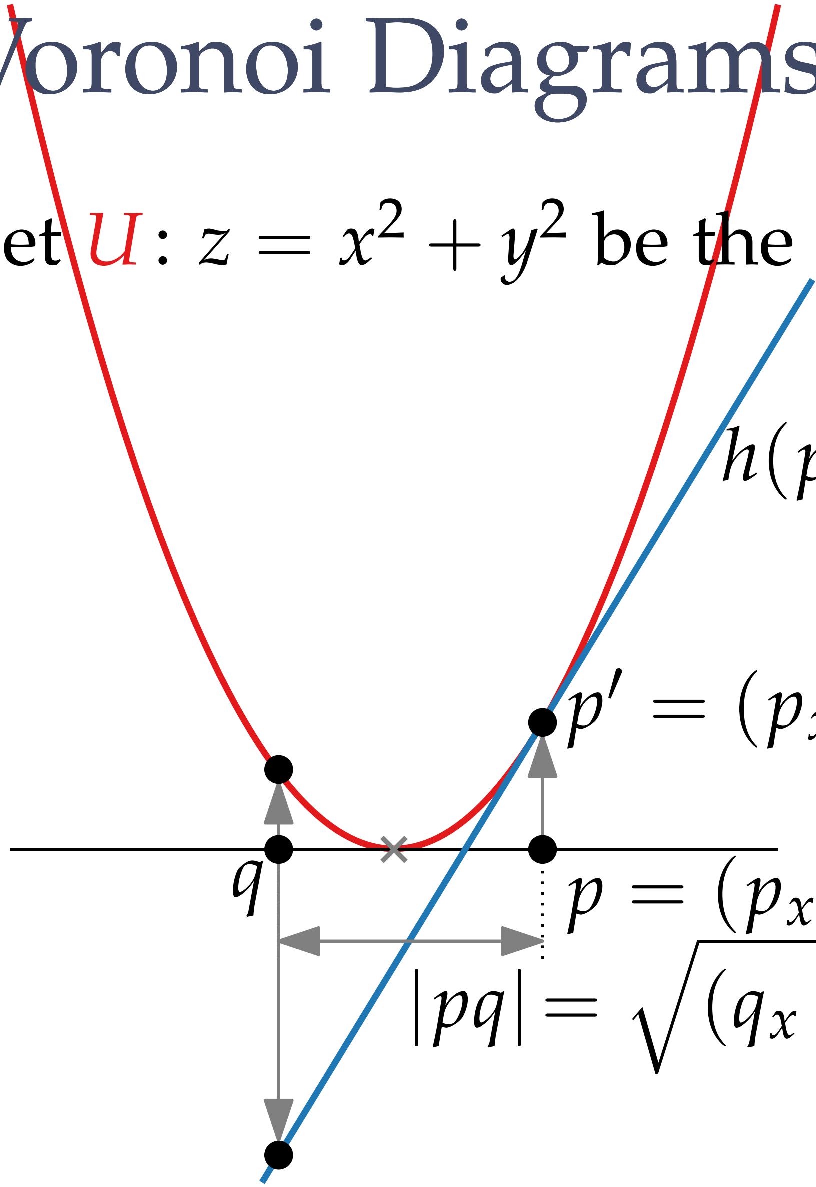
$p' = (p_x, p_y, p_x^2 + p_y^2)$

$p = (p_x, p_y, 0)$

$|pq| = \sqrt{(q_x - p_x)^2 + (q_y - p_y)^2}$

Voronoi Diagrams Revisited

Let $U\colon z = x^2 + y^2$ be the *unit paraboloid* in $\mathbb{R}^3$.



$$h(p)\colon z = (2p_x)x + (2p_y)y - (p_x^2 + p_y^2)$$

Note that $p' \in h(p)$.

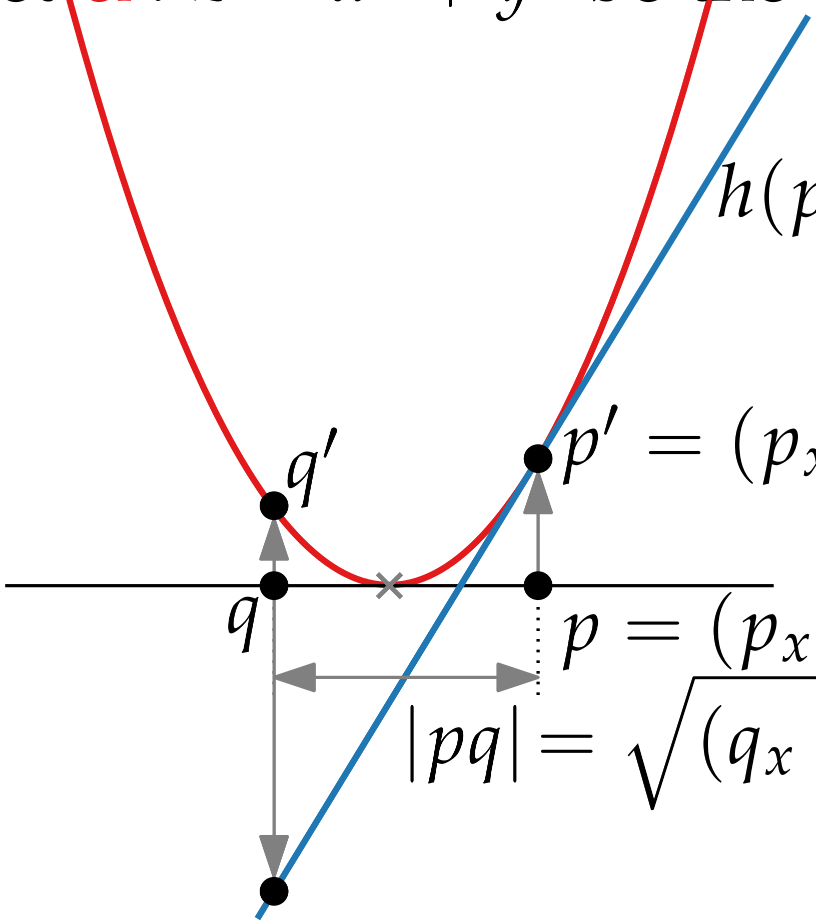
$$p' = (p_x, p_y, p_x^2 + p_y^2)$$

$$p = (p_x, p_y, 0)$$

$$|pq| = \sqrt{(q_x - p_x)^2 + (q_y - p_y)^2}$$

Voronoi Diagrams Revisited

Let $U\colon z = x^2 + y^2$ be the *unit paraboloid* in $\mathbb{R}^3$.



$h(p)\colon z = (2p_x)x + (2p_y)y - (p_x^2 + p_y^2)$

Note that $p' \in h(p)$.

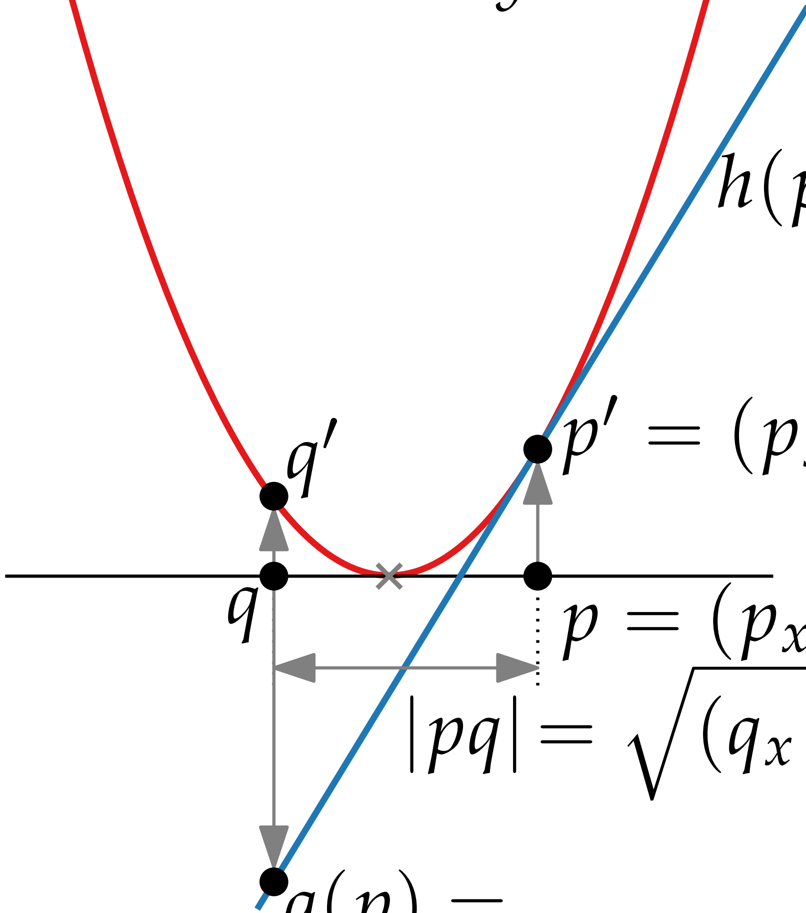
$p' = (p_x, p_y, p_x^2 + p_y^2)$

$p = (p_x, p_y, 0)$

$|pq| = \sqrt{(q_x - p_x)^2 + (q_y - p_y)^2}$

Voronoi Diagrams Revisited

Let $U\colon z = x^2 + y^2$ be the *unit paraboloid* in $\mathbb{R}^3$.



$h(p)\colon z = (2p_x)x + (2p_y)y - (p_x^2 + p_y^2)$

Note that $p' \in h(p)$.

$p' = (p_x, p_y, p_x^2 + p_y^2)$

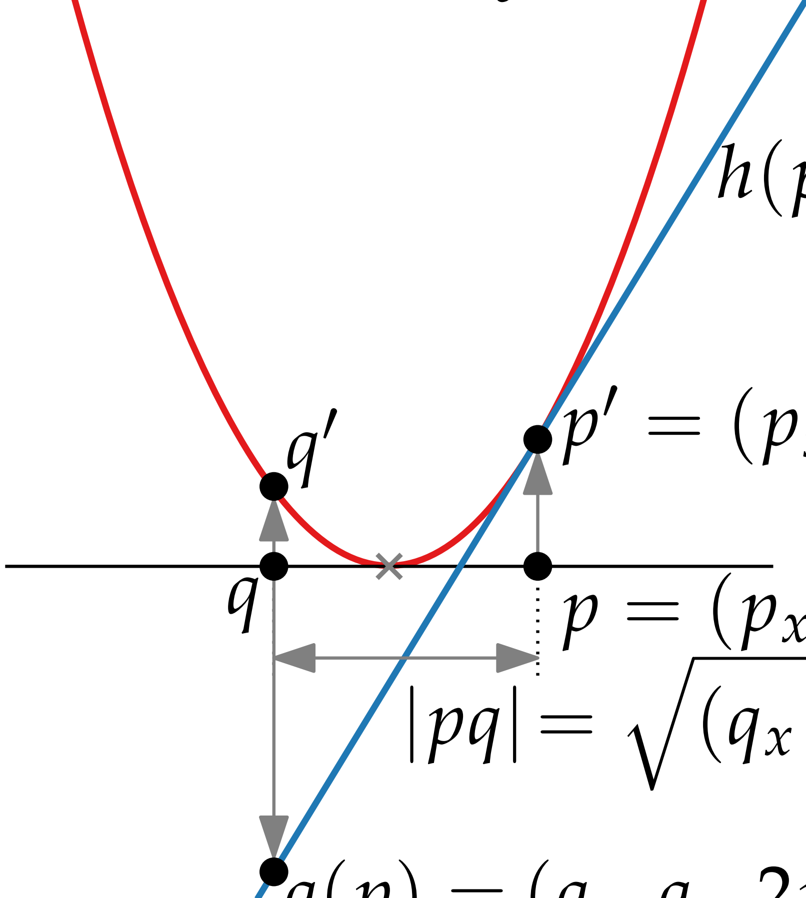
$p = (p_x, p_y, 0)$

$|pq| = \sqrt{(q_x - p_x)^2 + (q_y - p_y)^2}$

$q(p) =$

Voronoi Diagrams Revisited

Let $U\colon z = x^2 + y^2$ be the *unit paraboloid* in $\mathbb{R}^3$.



$h(p)\colon z = (2p_x)x + (2p_y)y - (p_x^2 + p_y^2)$

Note that $p' \in h(p)$.

$p' = (p_x, p_y, p_x^2 + p_y^2)$

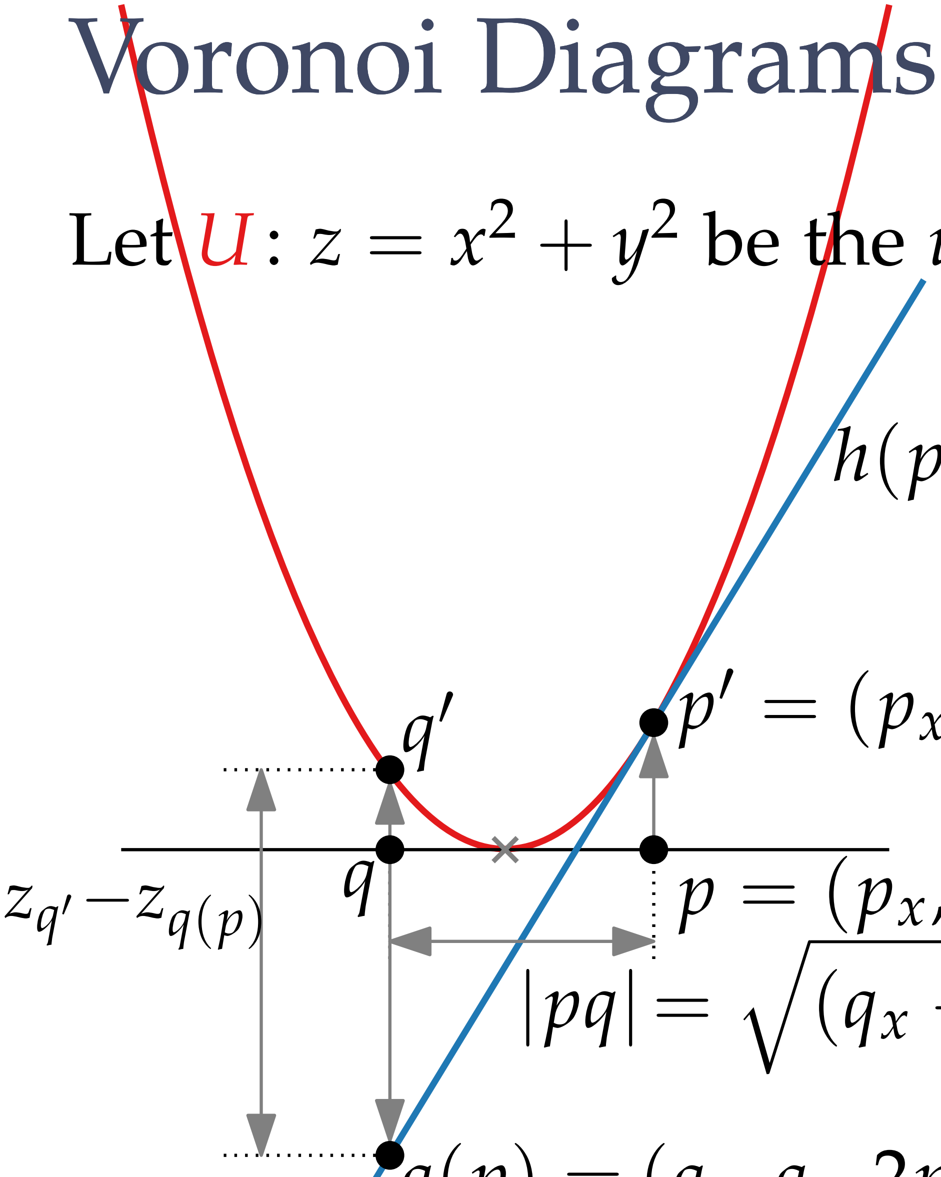
$p = (p_x, p_y, 0)$

$|pq| = \sqrt{(q_x - p_x)^2 + (q_y - p_y)^2}$

$q(p) = (q_x, q_y, 2p_xq_x + 2p_yq_y - (p_x^2 + p_y^2))$

Voronoi Diagrams Revisited

Let U: $z = x^2 + y^2$ be the *unit paraboloid* in $\mathbb{R}^3$.



$h(p)\colon z = (2p_x)x + (2p_y)y - (p_x^2 + p_y^2)$

Note that $p' \in h(p)$.

$p' = (p_x, p_y, p_x^2 + p_y^2)$

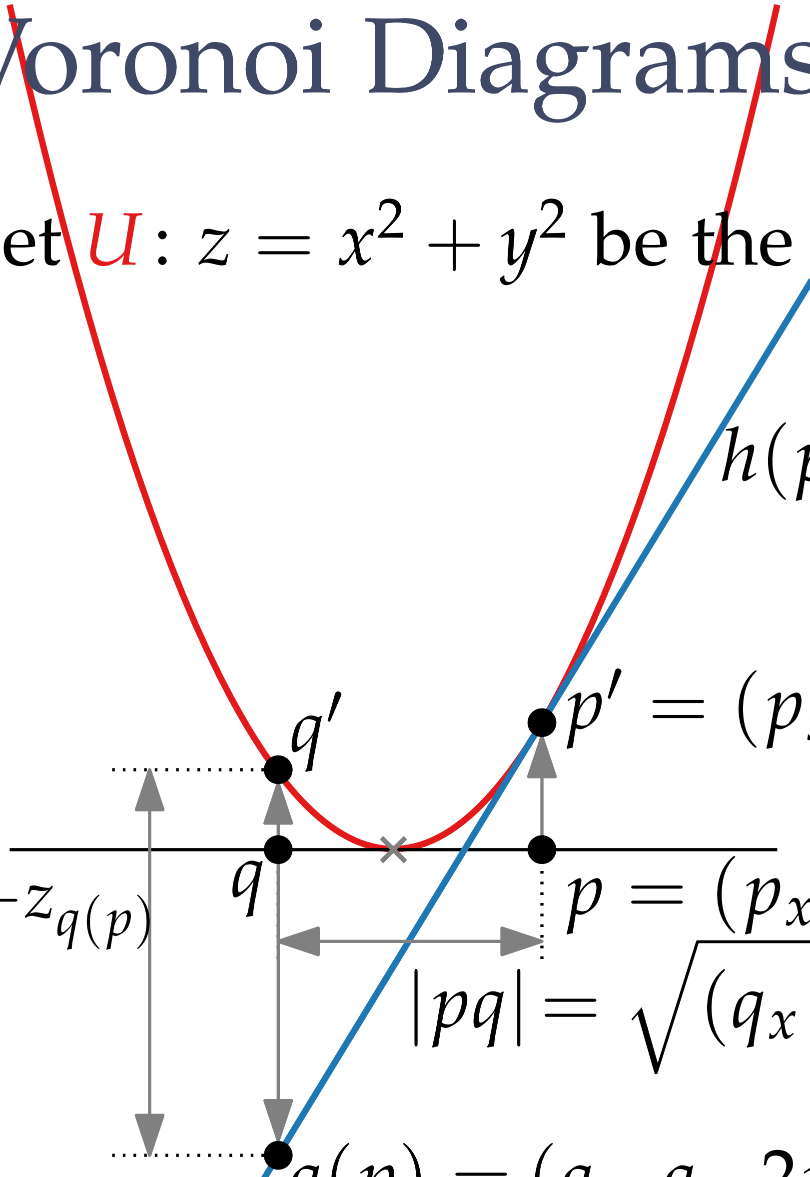
$p = (p_x, p_y, 0)$

$|pq| = \sqrt{(q_x - p_x)^2 + (q_y - p_y)^2}$

$q(p) = (q_x, q_y, 2p_xq_x + 2p_yq_y - (p_x^2 + p_y^2))$

Voronoi Diagrams Revisited

Let U: $z = x^2 + y^2$ be the *unit paraboloid* in $\mathbb{R}^3$.



$$h(p)\colon z = (2p_x)x + (2p_y)y - (p_x^2 + p_y^2)$$

Note that $p' \in h(p)$.

$$p' = (p_x, p_y, p_x^2 + p_y^2)$$

$$z_{q'} - z_{q(p)}$$

$$p = (p_x, p_y, 0)$$

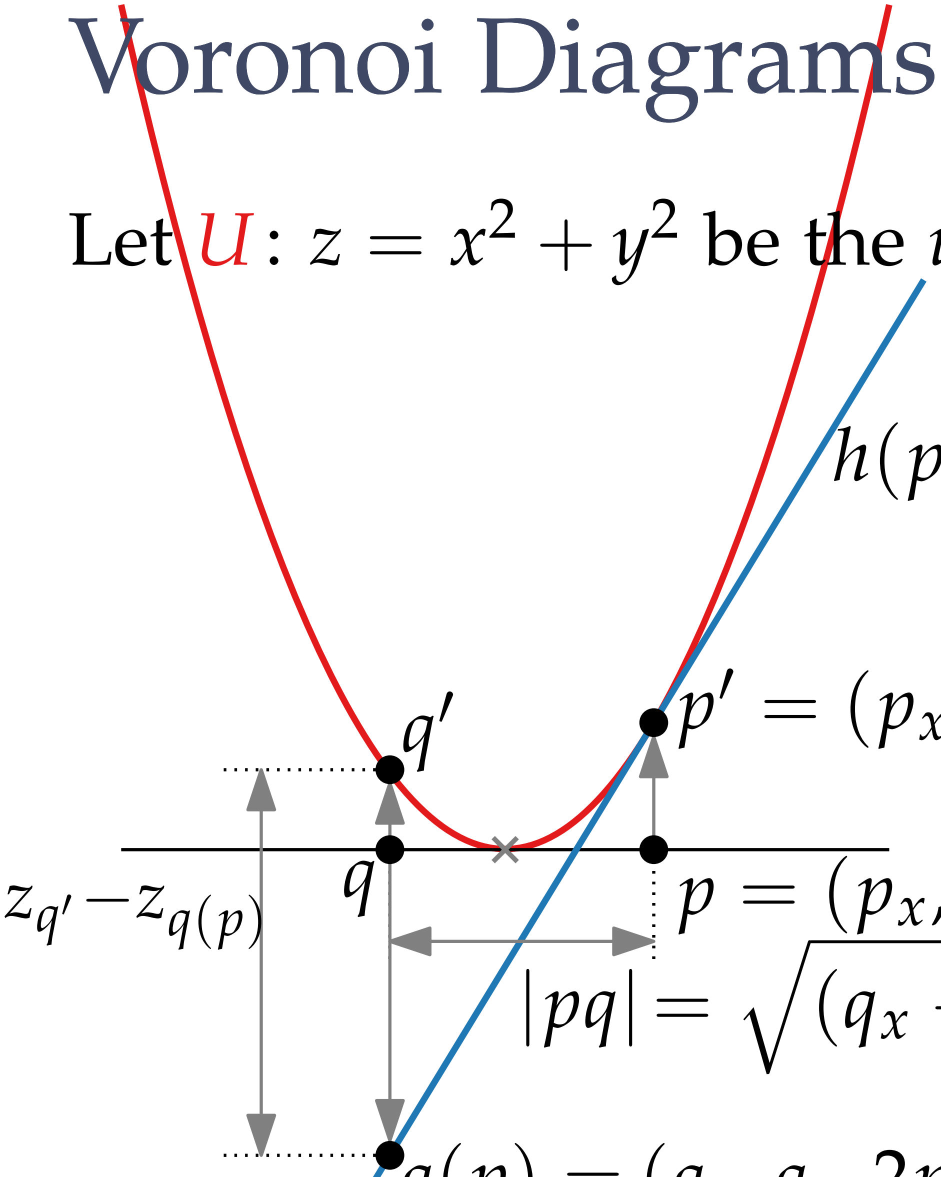
$$|pq| = \sqrt{(q_x - p_x)^2 + (q_y - p_y)^2}$$

$$q(p) = (q_x, q_y, 2p_xq_x + 2p_yq_y - (p_x^2 + p_y^2))$$

$$q_x^2 + q_y^2 - 2p_xq_x - 2p_yq_y + (p_x^2 + p_y^2)$$

Voronoi Diagrams Revisited

Let $U: z = x^2 + y^2$ be the *unit paraboloid* in $\mathbb{R}^3$.



$h(p): z = (2p_x)x + (2p_y)y - (p_x^2 + p_y^2)$

Note that $p' \in h(p)$.

$p' = (p_x, p_y, p_x^2 + p_y^2)$

$p = (p_x, p_y, 0)$

$|pq| = \sqrt{(q_x - p_x)^2 + (q_y - p_y)^2}$

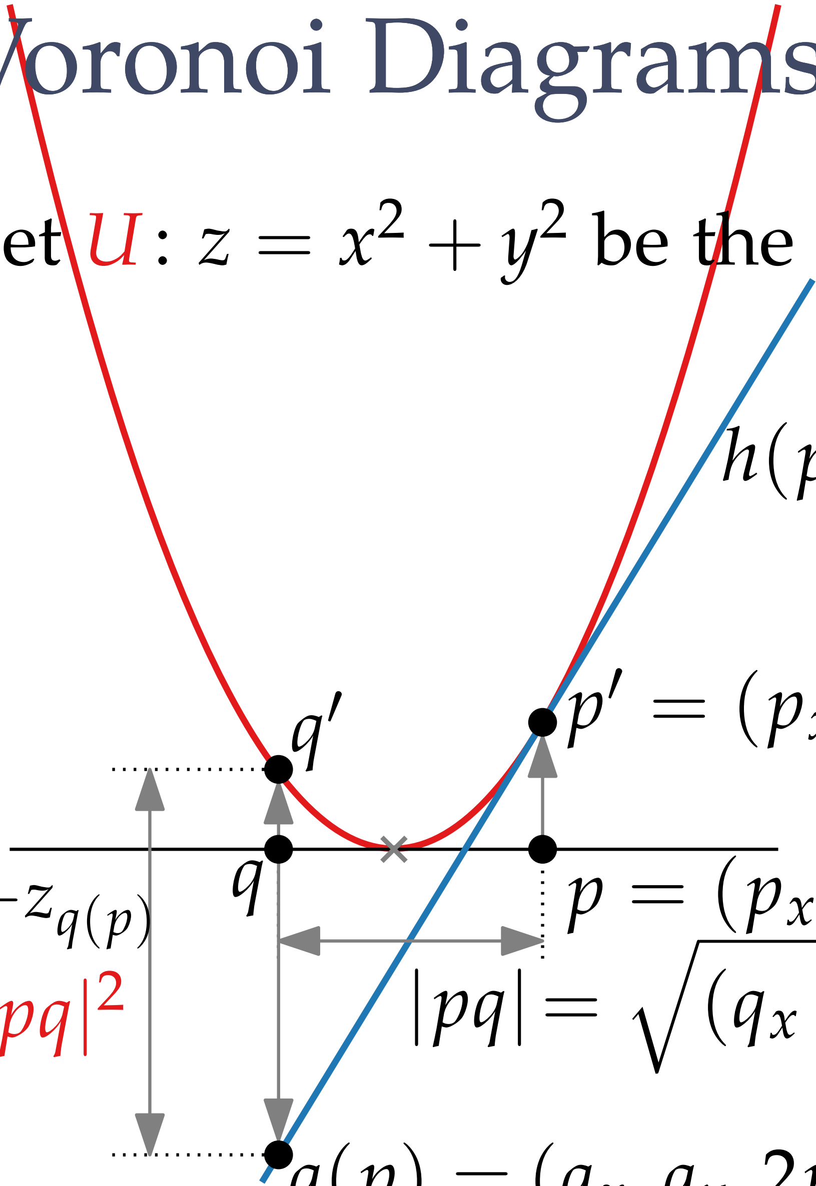
$q(p) = (q_x, q_y, 2p_xq_x + 2p_yq_y - (p_x^2 + p_y^2))$

$q_x^2 + q_y^2 - 2p_xq_x - 2p_yq_y + (p_x^2 + p_y^2)$

$(q_x^2 - 2p_xq_x + p_x^2) + (q_y^2 - 2p_yq_y + p_y^2)$

Voronoi Diagrams Revisited

Let $U: z = x^2 + y^2$ be the *unit paraboloid* in $\mathbb{R}^3$.



$$h(p): z = (2p_x)x + (2p_y)y - (p_x^2 + p_y^2)$$

Note that $p' \in h(p)$.

$$p' = (p_x, p_y, p_x^2 + p_y^2)$$

$$p = (p_x, p_y, 0)$$

$$z_{q'} - z_{q(p)} = |pq|^2$$

$$|pq| = \sqrt{(q_x - p_x)^2 + (q_y - p_y)^2}$$

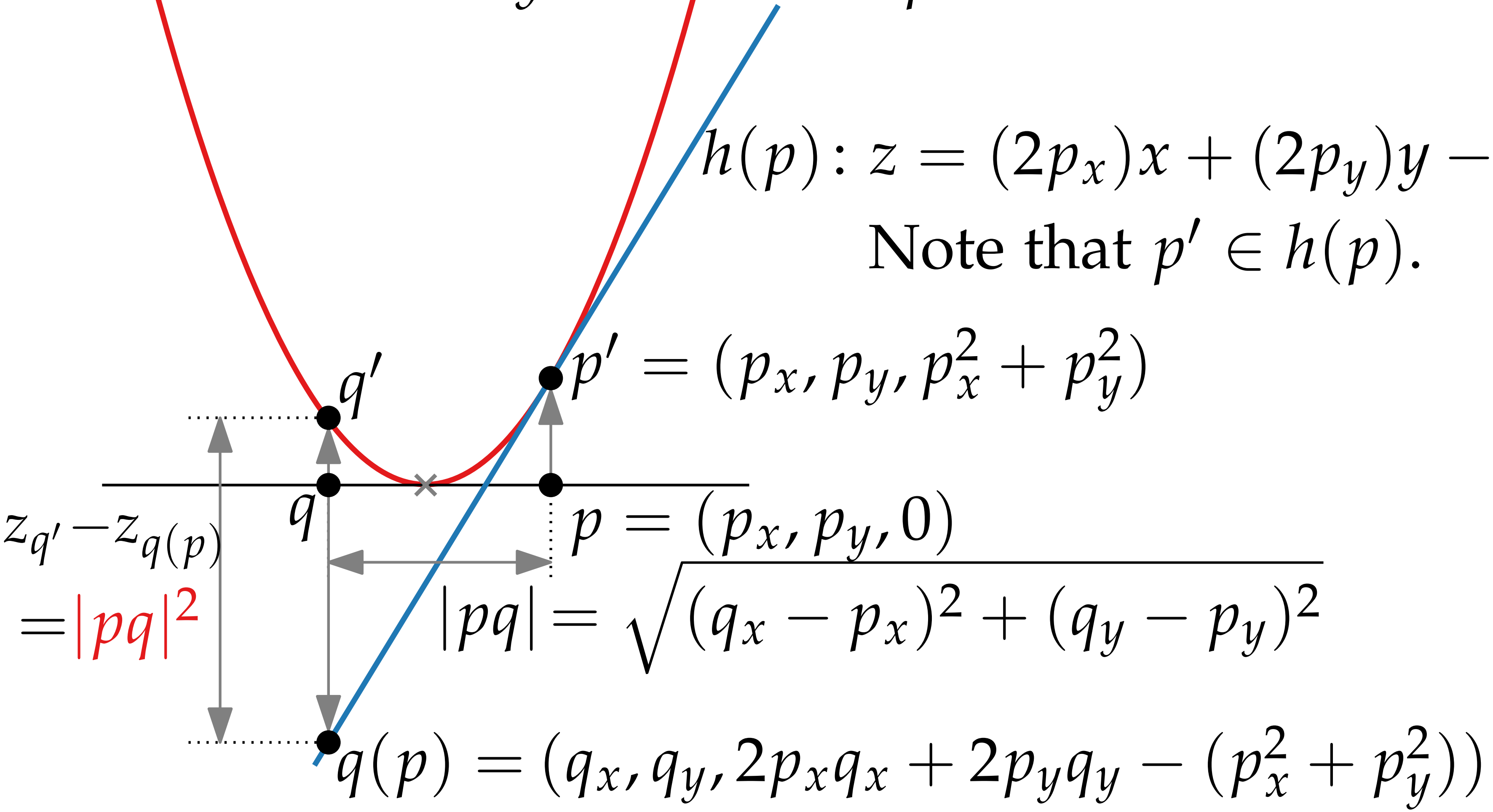
$$q(p) = (q_x, q_y, 2p_xq_x + 2p_yq_y - (p_x^2 + p_y^2))$$

$$q_x^2 + q_y^2 - 2p_xq_x - 2p_yq_y + (p_x^2 + p_y^2)$$

$$(q_x^2 - 2p_xq_x + p_x^2) + (q_y^2 - 2p_yq_y + p_y^2)$$

Voronoi Diagrams Revisited

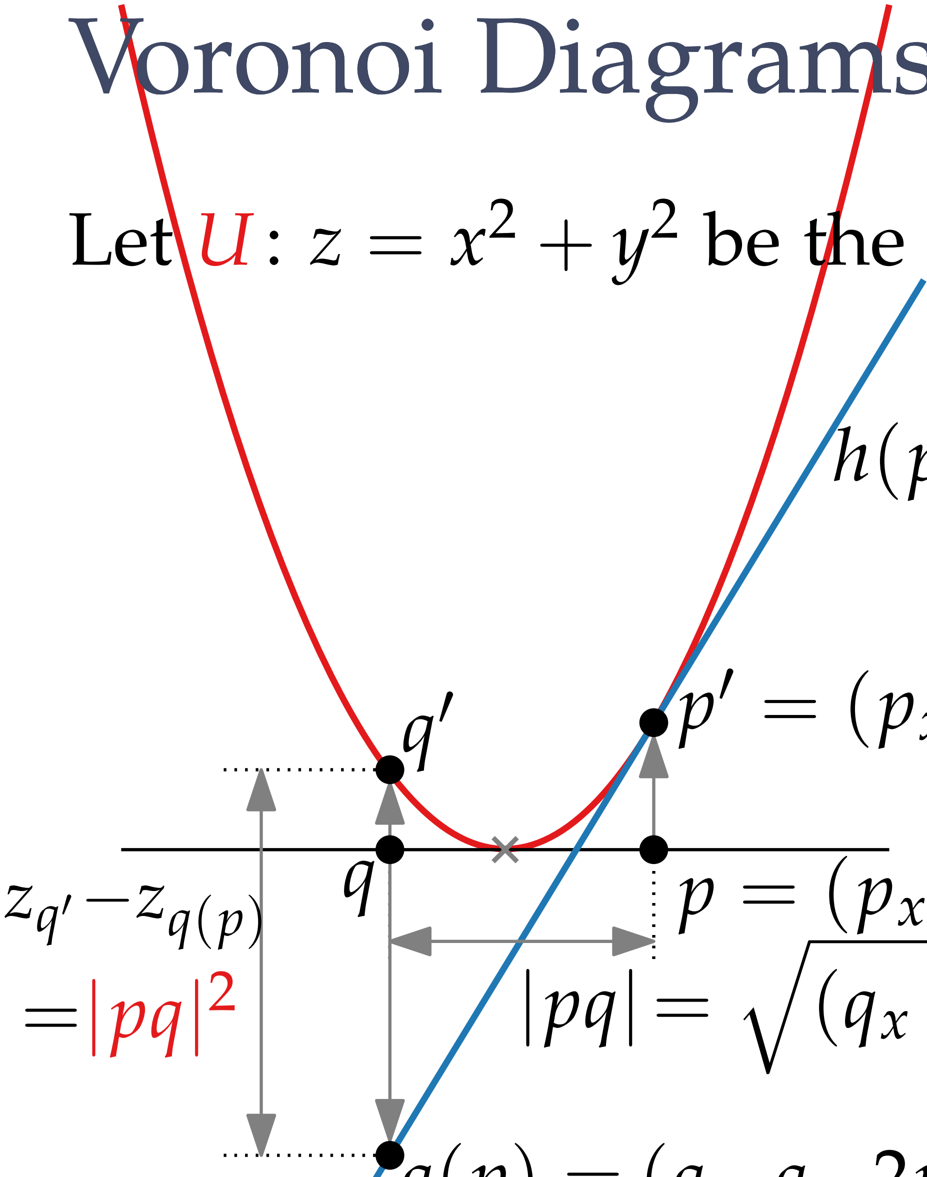
Let $U\colon z = x^2 + y^2$ be the *unit paraboloid* in $\mathbb{R}^3$.



$\Rightarrow h(p)$ and U encode dist. betw. p and any other pt in $z{=}0$.

Voronoi Diagrams Revisited

Let $U: z = x^2 + y^2$ be the *unit paraboloid* in $\mathbb{R}^3$.



$h(p): z = (2p_x)x + (2p_y)y - (p_x^2 + p_y^2)$

Note that $p' \in h(p)$.

$p' = (p_x, p_y, p_x^2 + p_y^2)$

$p = (p_x, p_y, 0)$

$|pq| = \sqrt{(q_x - p_x)^2 + (q_y - p_y)^2}$

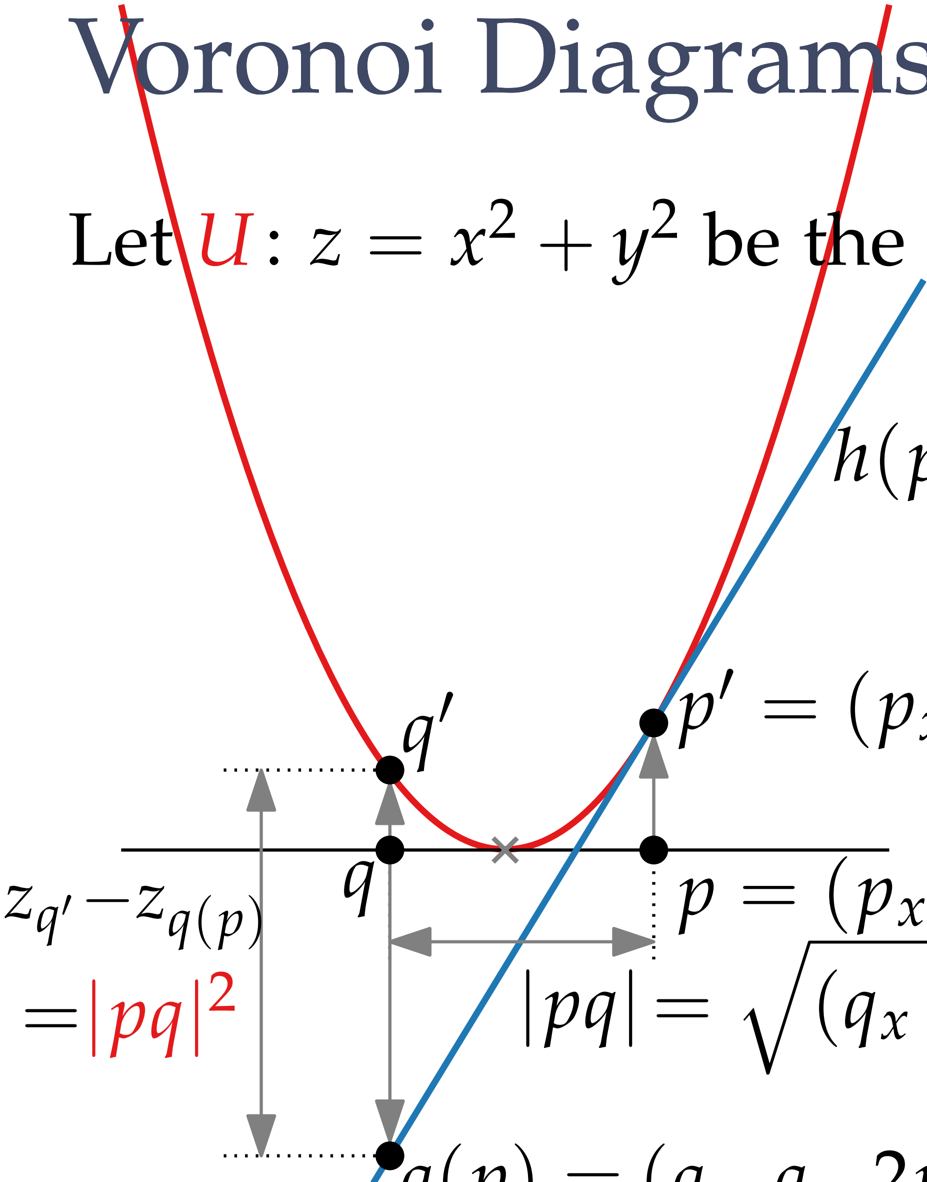
$q(p) = (q_x, q_y, 2p_xq_x + 2p_yq_y - (p_x^2 + p_y^2))$

$\Rightarrow h(p)$ and U encode dist. betw. p and any other pt in $z=0$.

$\Rightarrow h(p) \cap U = \{p'\}$

Voronoi Diagrams Revisited

Let $U\colon z = x^2 + y^2$ be the *unit paraboloid* in $\mathbb{R}^3$.



$h(p)\colon z = (2p_x)x + (2p_y)y - (p_x^2 + p_y^2)$

Note that $p' \in h(p)$.

$p' = (p_x, p_y, p_x^2 + p_y^2)$

$p = (p_x, p_y, 0)$

$|pq| = \sqrt{(q_x - p_x)^2 + (q_y - p_y)^2}$

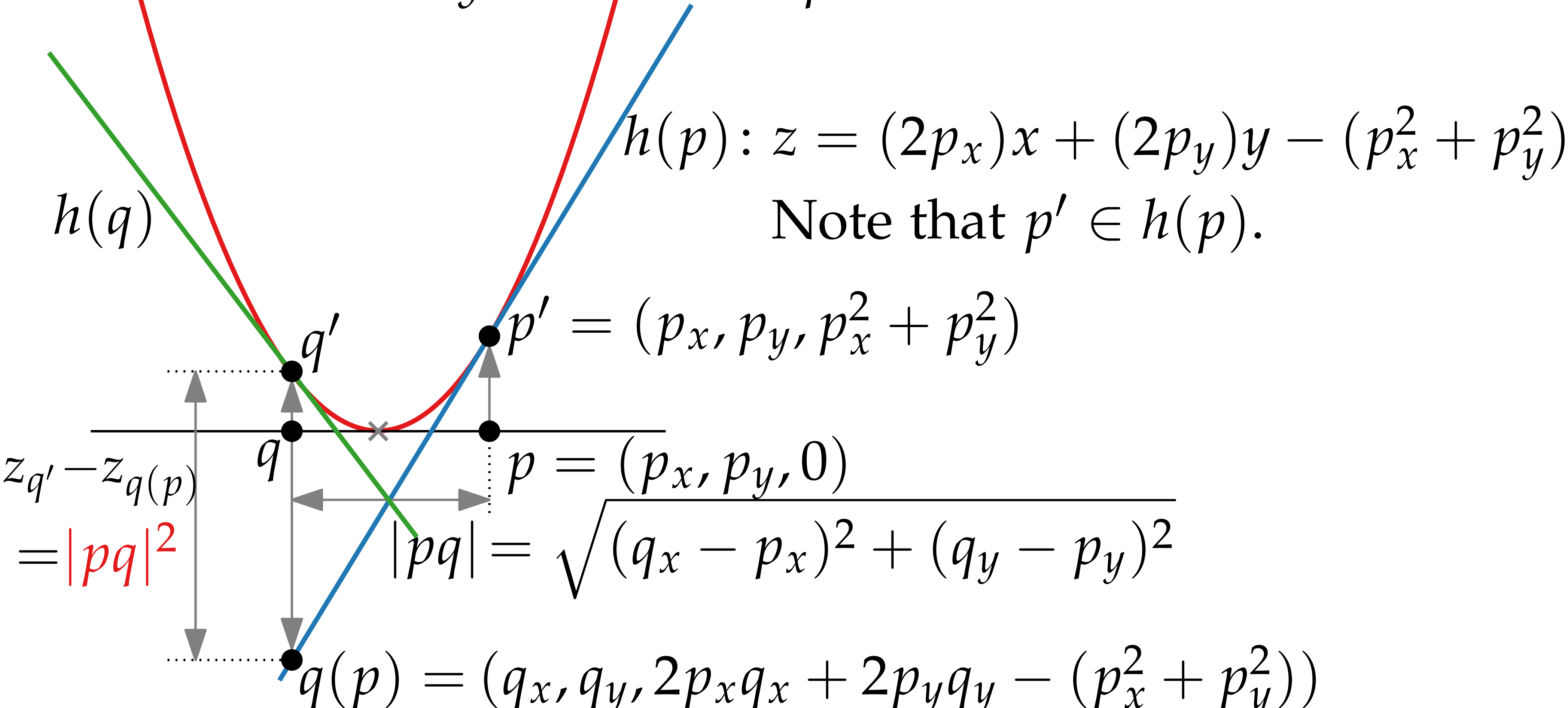
$q(p) = (q_x, q_y, 2p_xq_x + 2p_yq_y - (p_x^2 + p_y^2))$

$\Rightarrow$ $h(p)$ and U encode dist. betw. p and any other pt in $z{=}0$.

$\Rightarrow$ $h(p) \cap U = \{p'\}$ $\;\Rightarrow\;$ $h(p)$ is tangent to U (in p')

Voronoi Diagrams Revisited

Let $U\colon z = x^2 + y^2$ be the *unit paraboloid* in $\mathbb{R}^3$.

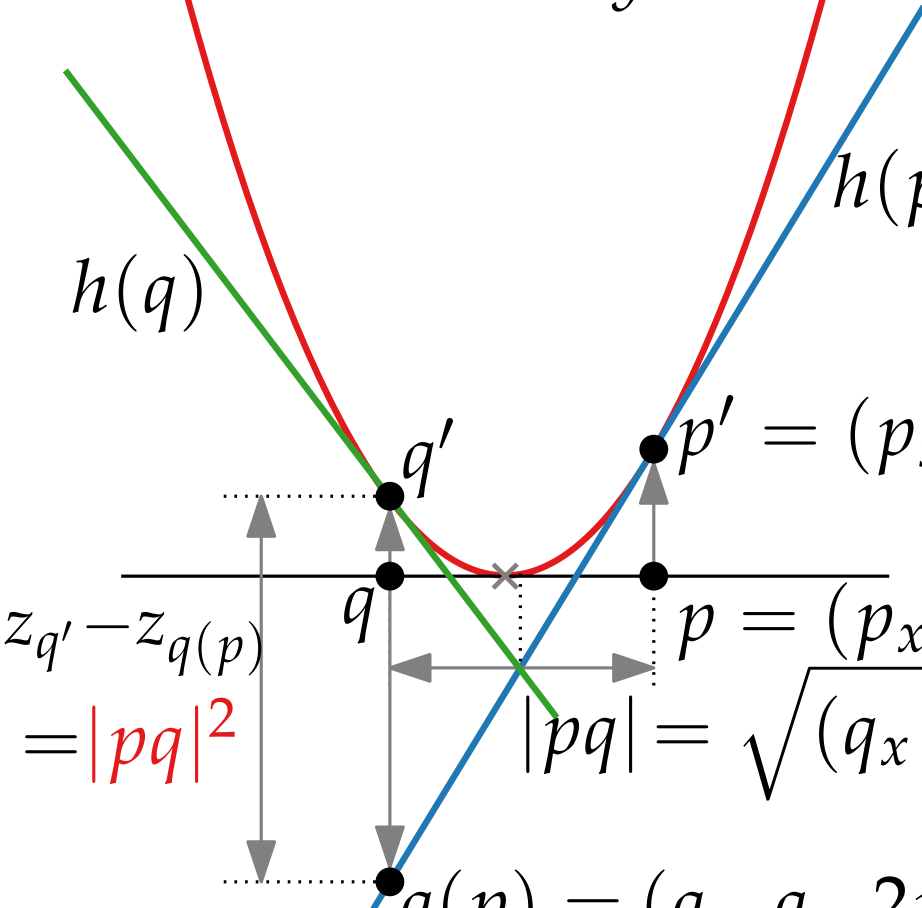


$\Rightarrow$ $h(p)$ and U encode dist. betw. p and any other pt in $z{=}0$.

$\Rightarrow$ $h(p) \cap U = \{p'\}$ $\Rightarrow$ $h(p)$ is tangent to U (in p')

Voronoi Diagrams Revisited

Let $U\colon z = x^2 + y^2$ be the *unit paraboloid* in $\mathbb{R}^3$.



$h(p)\colon z = (2p_x)x + (2p_y)y - (p_x^2 + p_y^2)$

Note that $p' \in h(p)$.

$p' = (p_x, p_y, p_x^2 + p_y^2)$

$p = (p_x, p_y, 0)$

$|pq| = \sqrt{(q_x - p_x)^2 + (q_y - p_y)^2}$

$q(p) = (q_x, q_y, 2p_xq_x + 2p_yq_y - (p_x^2 + p_y^2))$

$\Rightarrow$ $h(p)$ and U encode dist. betw. p and any other pt in $z{=}0$.

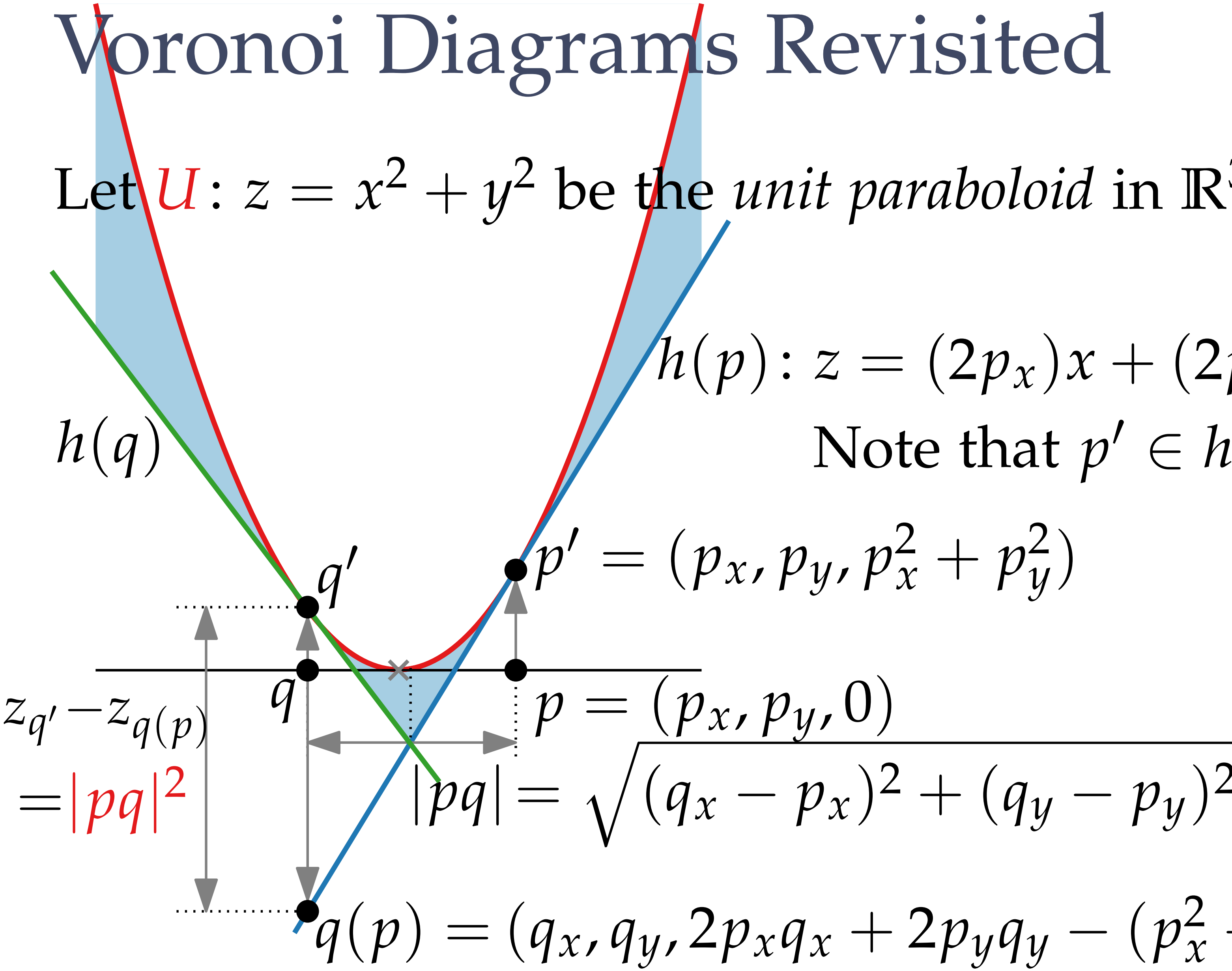
$\Rightarrow$ $h(p) \cap U = \{p'\}$ $\Rightarrow$ $h(p)$ is tangent to U (in p')

Voronoi Diagrams Revisited

Let $U\colon z = x^2 + y^2$ be the *unit paraboloid* in $\mathbb{R}^3$.

$h(p)\colon z = (2p_x)x + (2p_y)y - (p_x^2 + p_y^2)$

Note that $p' \in h(p)$.



$\Rightarrow h(p)$ and U encode dist. betw. p and any other pt in $z{=}0$.

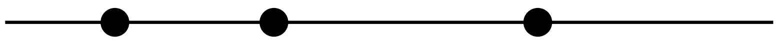
$\Rightarrow h(p) \cap U = \{p'\} \;\Rightarrow\; h(p)$ is tangent to U (in p')

The Upper Envelope Strikes Back

Theorem. Let $P \subset \mathbb{R}^2 \times \{0\}$

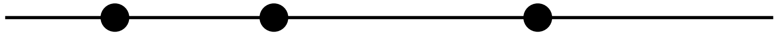
The Upper Envelope Strikes Back

Theorem. Let $P \subset \mathbb{R}^2 \times \{0\}$



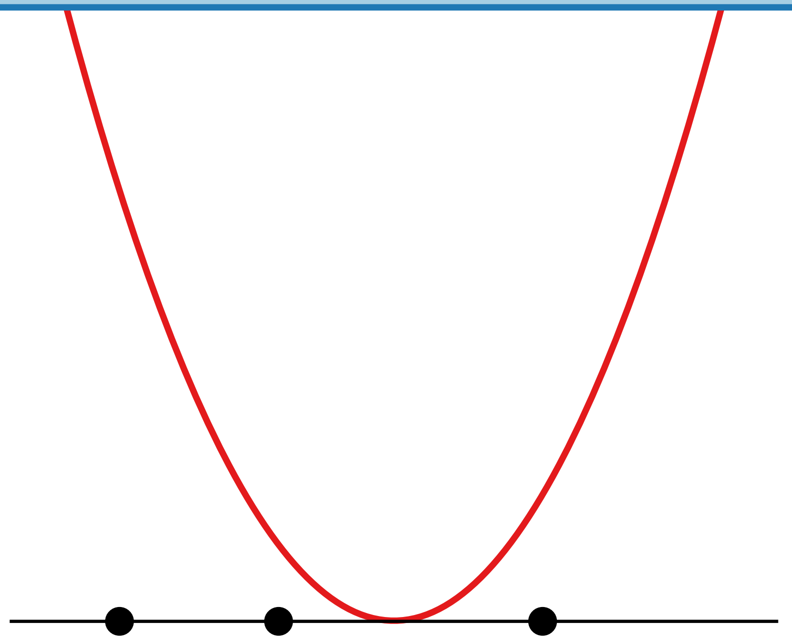
The Upper Envelope Strikes Back

Theorem. Let $P \subset \mathbb{R}^2 \times \{0\}$ and $\mathcal{H} = \{h(p) \mid p \in P\}$.



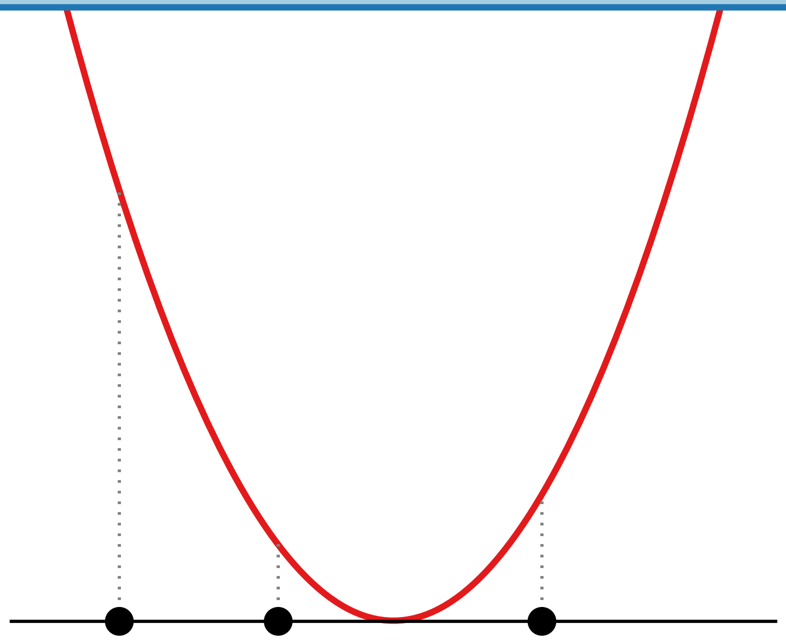
The Upper Envelope Strikes Back

Theorem. Let $P \subset \mathbb{R}^2 \times \{0\}$ and $\mathcal{H} = \{h(p) \mid p \in P\}$.



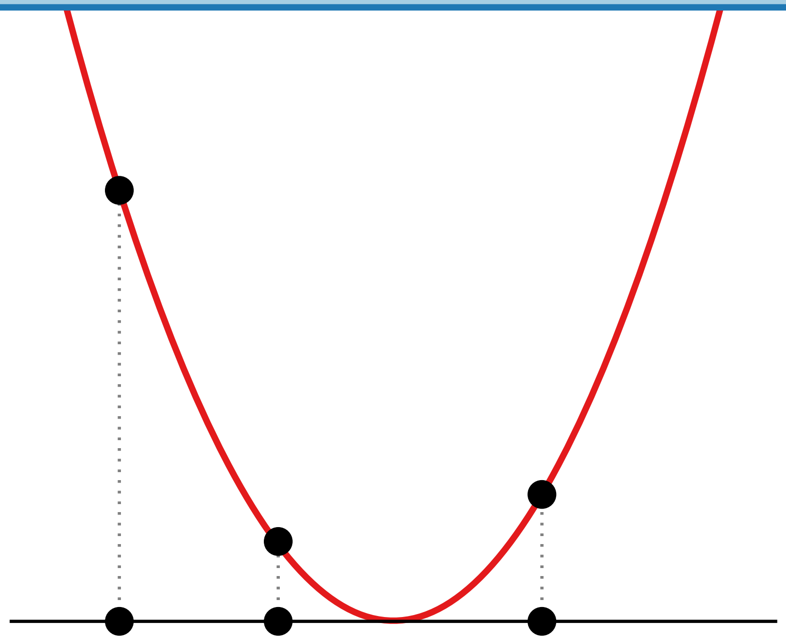
The Upper Envelope Strikes Back

Theorem. Let $P \subset \mathbb{R}^2 \times \{0\}$ and $\mathcal{H} = \{h(p) \mid p \in P\}$.



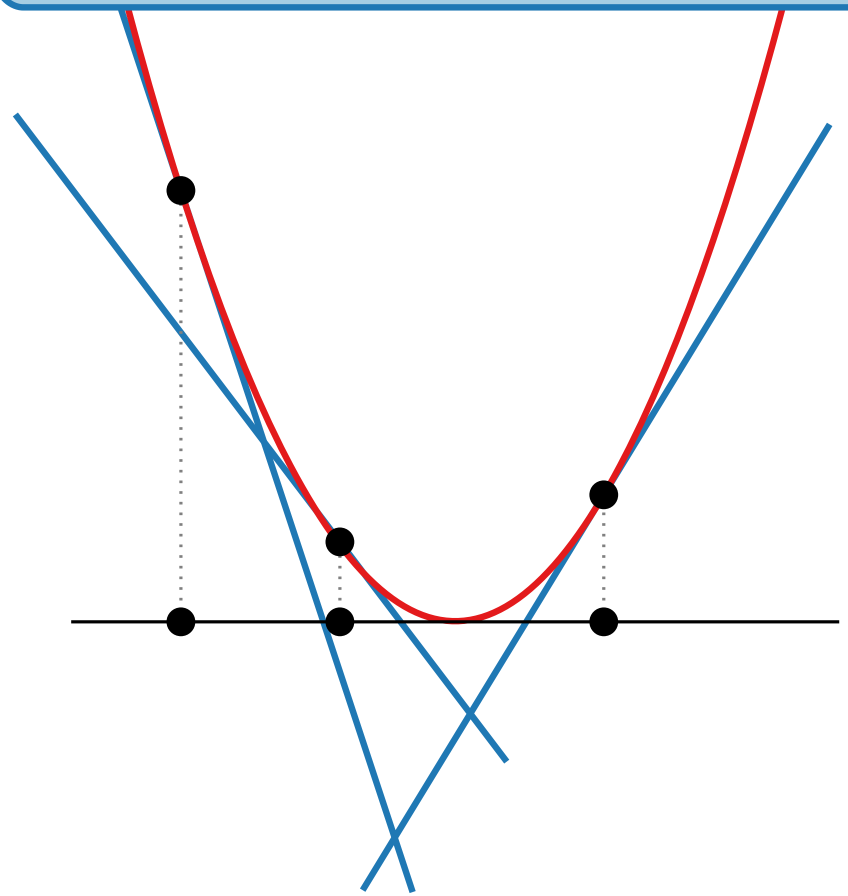
The Upper Envelope Strikes Back

Theorem. Let $P \subset \mathbb{R}^2 \times \{0\}$ and $\mathcal{H} = \{h(p) \mid p \in P\}$.



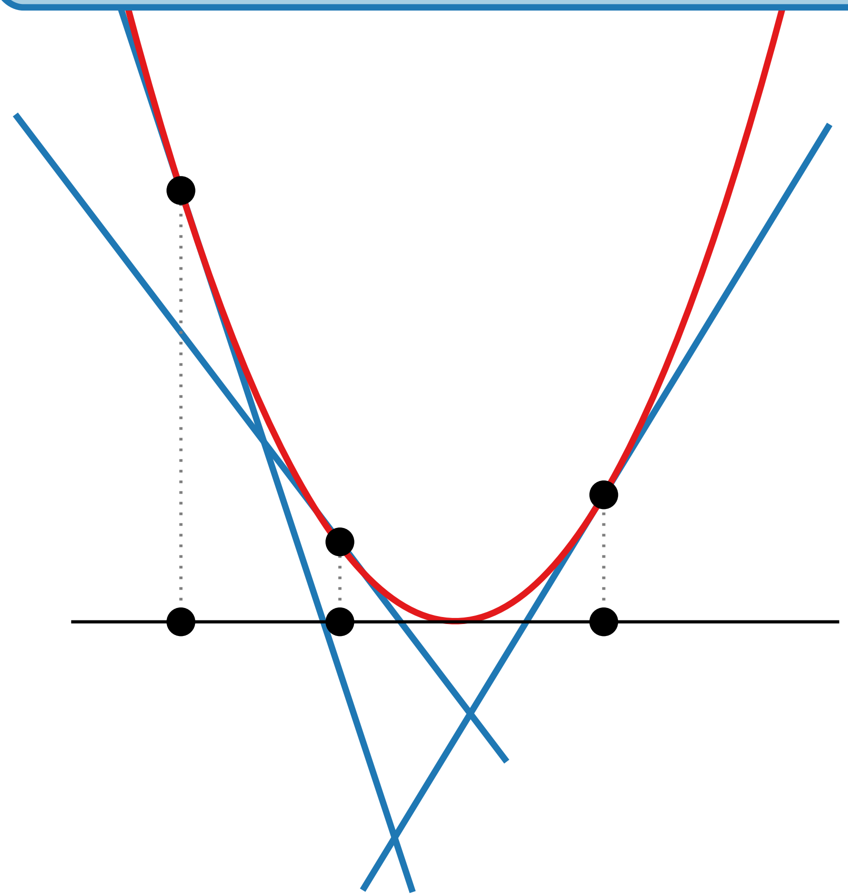
The Upper Envelope Strikes Back

Theorem. Let $P \subset \mathbb{R}^2 \times \{0\}$ and $\mathcal{H} = \{h(p) \mid p \in P\}$.



The Upper Envelope Strikes Back

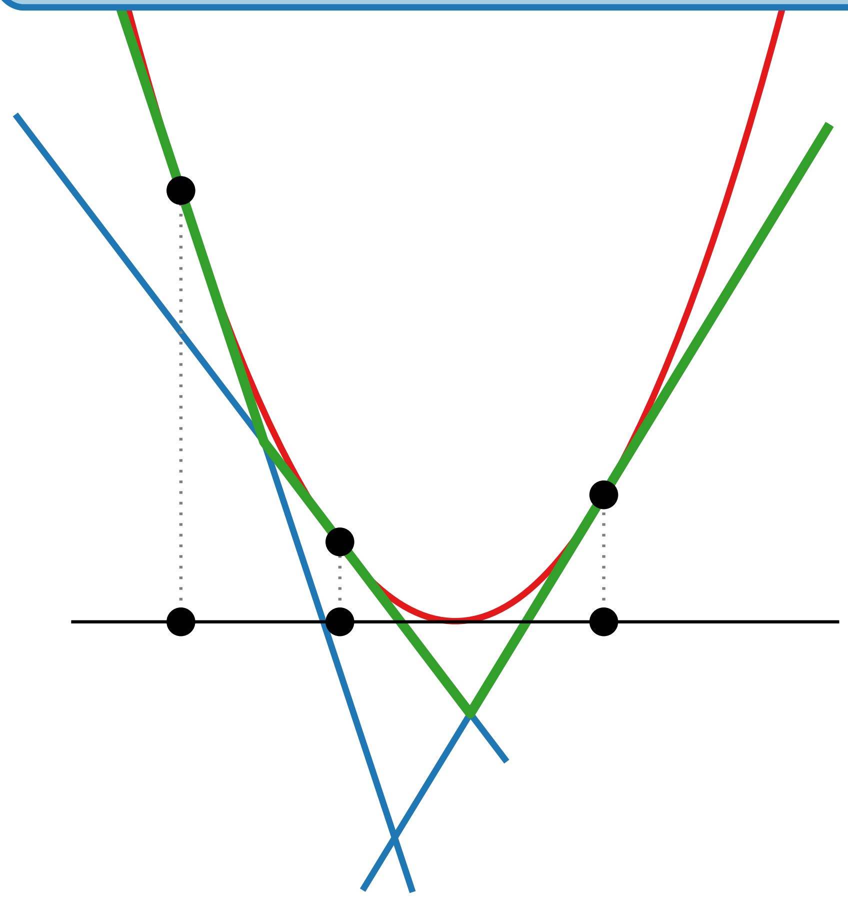
Theorem. Let $P \subset \mathbb{R}^2 \times \{0\}$ and $\mathcal{H} = \{h(p) \mid p \in P\}$.
Let $\mathcal{E}(\mathcal{H})$ be the upper envelope of $\mathcal{H}$.



The Upper Envelope Strikes Back

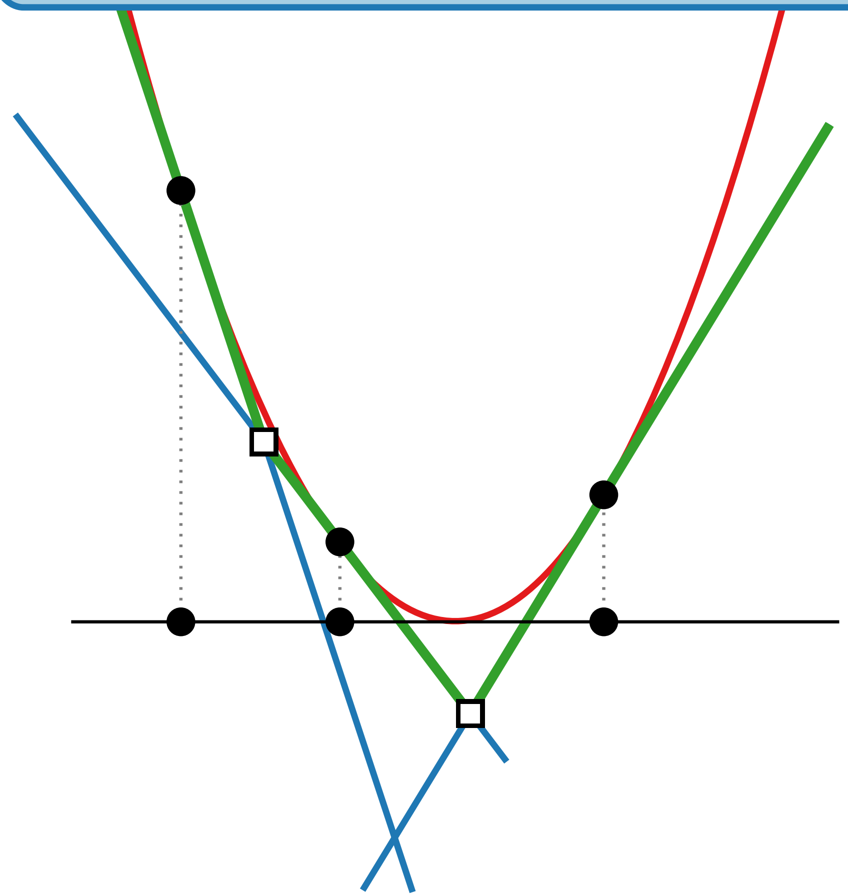
Theorem. Let $P \subset \mathbb{R}^2 \times \{0\}$ and $\mathcal{H} = \{h(p) \mid p \in P\}$.

Let $\mathcal{E}(\mathcal{H})$ be the upper envelope of $\mathcal{H}$.



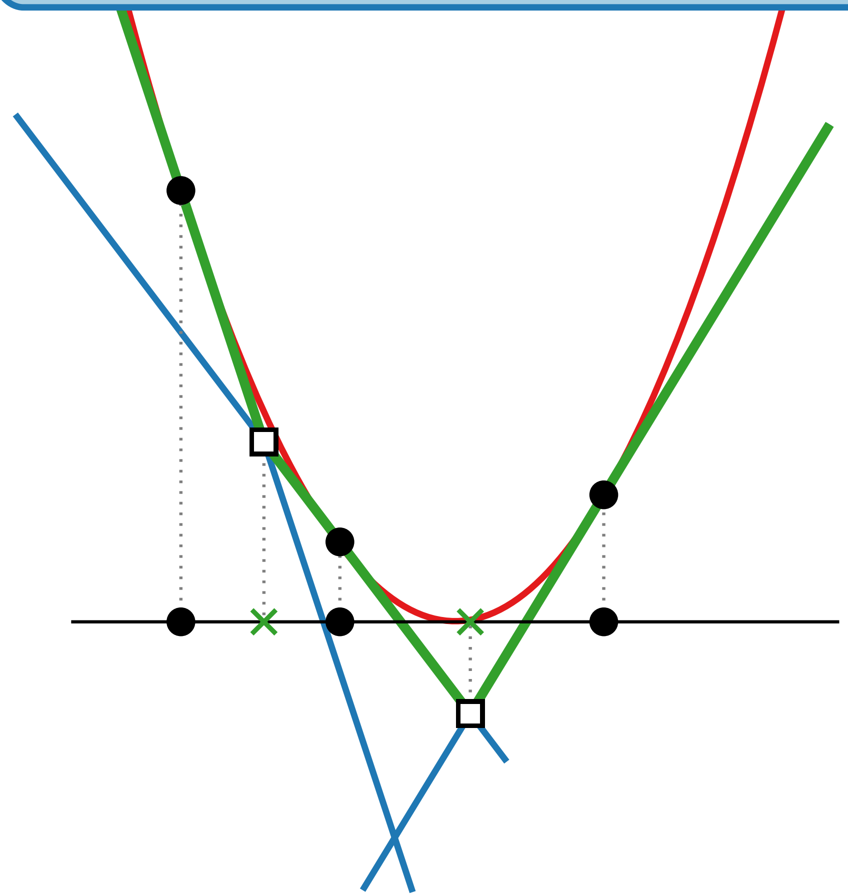
The Upper Envelope Strikes Back

Theorem. Let $P \subset \mathbb{R}^2 \times \{0\}$ and $\mathcal{H} = \{h(p) \mid p \in P\}$.
Let $\mathcal{E}(\mathcal{H})$ be the upper envelope of $\mathcal{H}$.



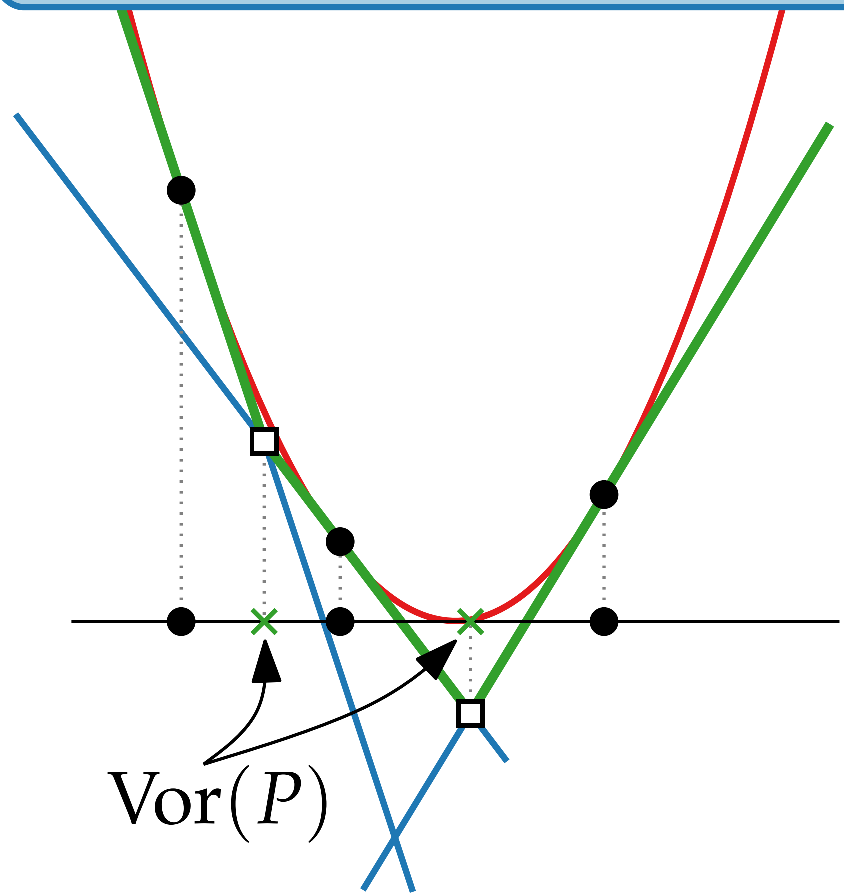
The Upper Envelope Strikes Back

Theorem. Let $P \subset \mathbb{R}^2 \times \{0\}$ and $\mathcal{H} = \{h(p) \mid p \in P\}$.
Let $\mathcal{E}(\mathcal{H})$ be the upper envelope of $\mathcal{H}$.
The projection of $\mathcal{E}(\mathcal{H})$ on $z = 0$ is



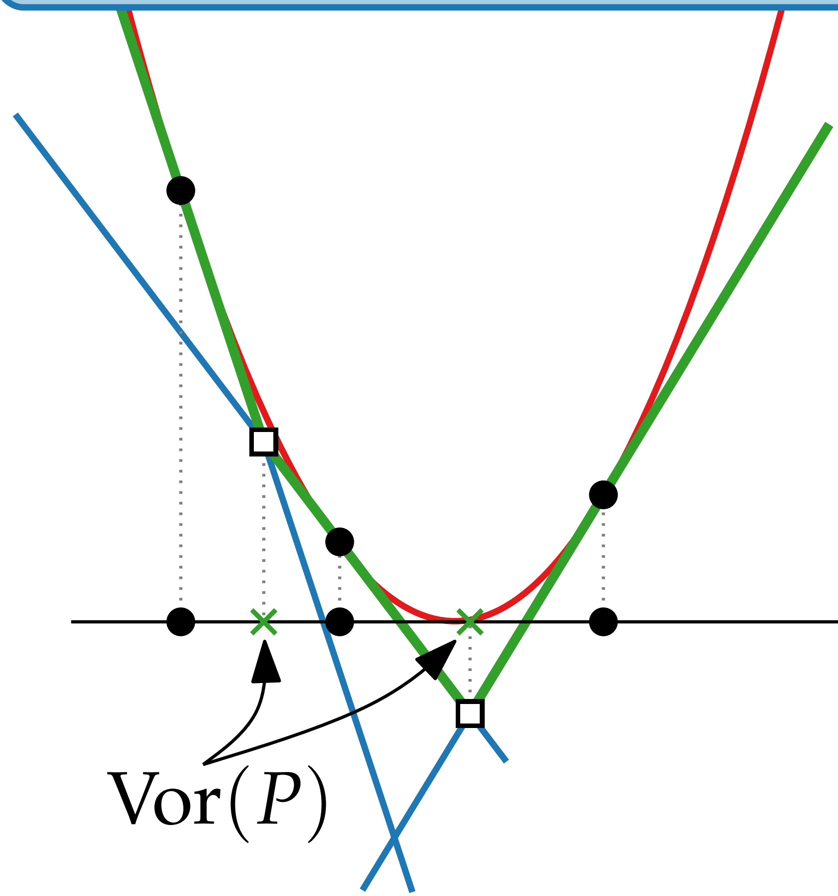
The Upper Envelope Strikes Back

Theorem. Let $P \subset \mathbb{R}^2 \times \{0\}$ and $\mathcal{H} = \{h(p) \mid p \in P\}$.
Let $\mathcal{E}(\mathcal{H})$ be the upper envelope of $\mathcal{H}$.
The projection of $\mathcal{E}(\mathcal{H})$ on $z = 0$ is $\mathrm{Vor}(P)$.



The Upper Envelope Strikes Back

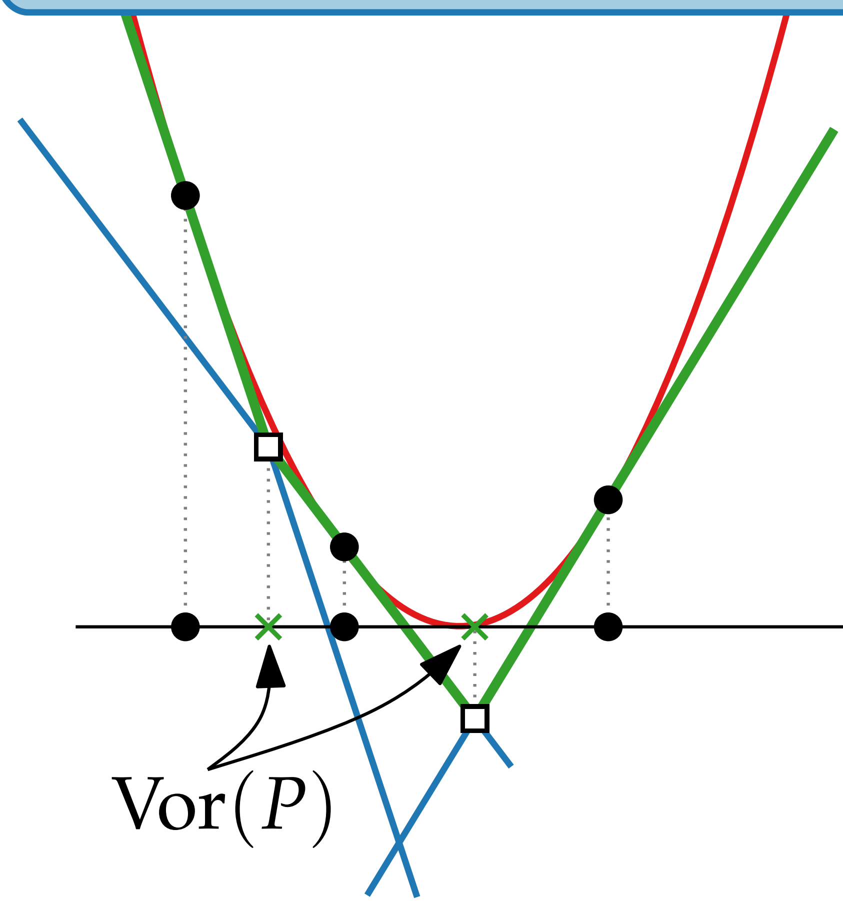
Theorem. Let $P \subset \mathbb{R}^2 \times \{0\}$ and $\mathcal{H} = \{h(p) \mid p \in P\}$.
Let $\mathcal{E}(\mathcal{H})$ be the upper envelope of $\mathcal{H}$.
The projection of $\mathcal{E}(\mathcal{H})$ on $z = 0$ is $\mathrm{Vor}(P)$.



can compute $\mathrm{Vor}(P)$ in $\mathbb{R}^2$
via upper envelope in $\mathbb{R}^3$

The Upper Envelope Strikes Back

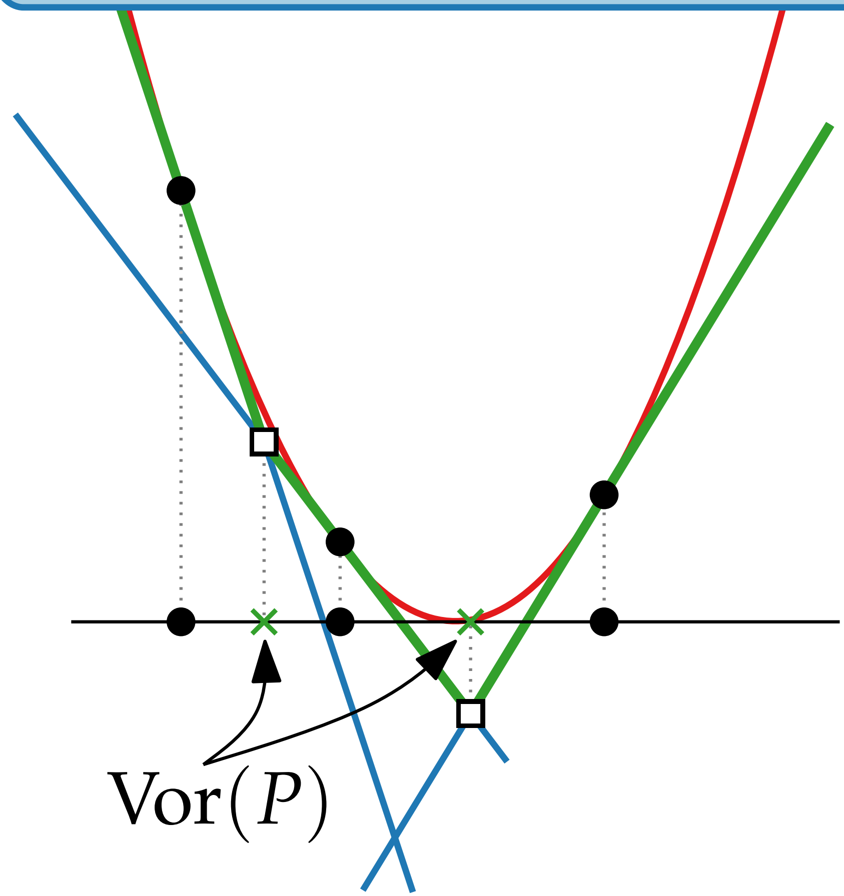
Theorem. Let $P \subset \mathbb{R}^2 \times \{0\}$ and $\mathcal{H} = \{h(p) \mid p \in P\}$.
Let $\mathcal{E}(\mathcal{H})$ be the upper envelope of $\mathcal{H}$.
The projection of $\mathcal{E}(\mathcal{H})$ on $z = 0$ is $\mathrm{Vor}(P)$.



can compute $\mathrm{Vor}(P)$ in $\mathbb{R}^2$
via upper envelope in $\mathbb{R}^3$

The Upper Envelope Strikes Back

Theorem. Let $P \subset \mathbb{R}^2 \times \{0\}$ and $\mathcal{H} = \{h(p) \mid p \in P\}$.
Let $\mathcal{E}(\mathcal{H})$ be the upper envelope of $\mathcal{H}$.
The projection of $\mathcal{E}(\mathcal{H})$ on $z = 0$ is $\text{Vor}(P)$.

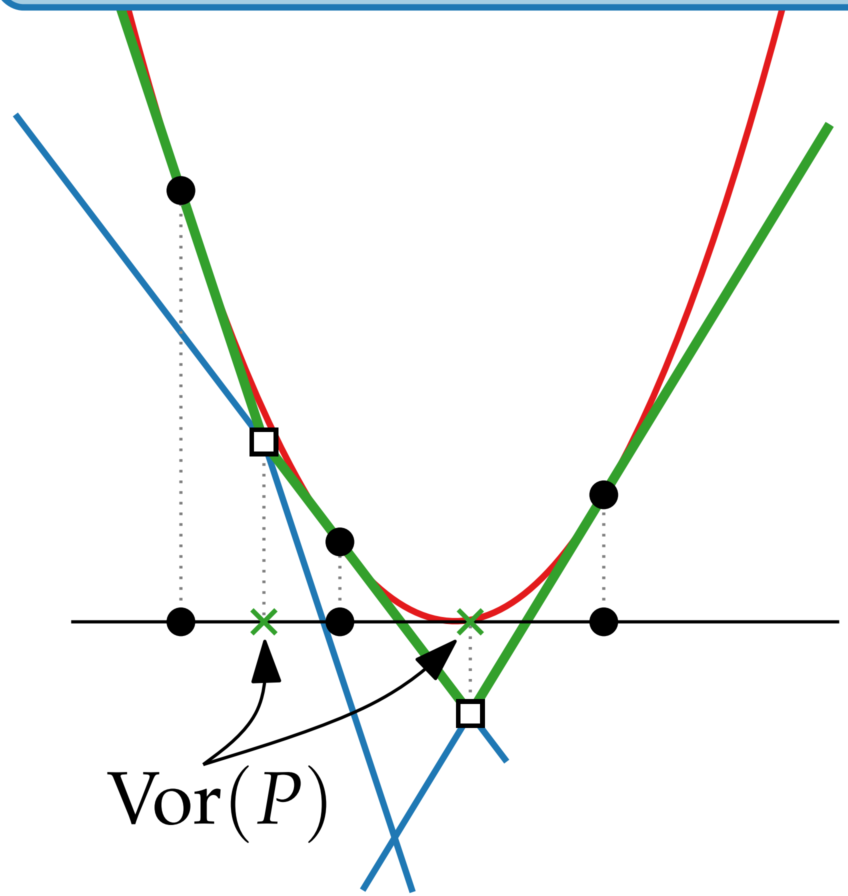


can compute $\text{Vor}(P)$ in $\mathbb{R}^2$ via upper envelope in $\mathbb{R}^3$

upper envelope in $\mathbb{R}^3$ is in one-to-one correspondence to lower convex hull of pt set $\mathcal{H}^\star$

The Upper Envelope Strikes Back

Theorem. Let $P \subset \mathbb{R}^2 \times \{0\}$ and $\mathcal{H} = \{h(p) \mid p \in P\}$.
Let $\mathcal{E}(\mathcal{H})$ be the upper envelope of $\mathcal{H}$.
The projection of $\mathcal{E}(\mathcal{H})$ on $z = 0$ is $\mathrm{Vor}(P)$.

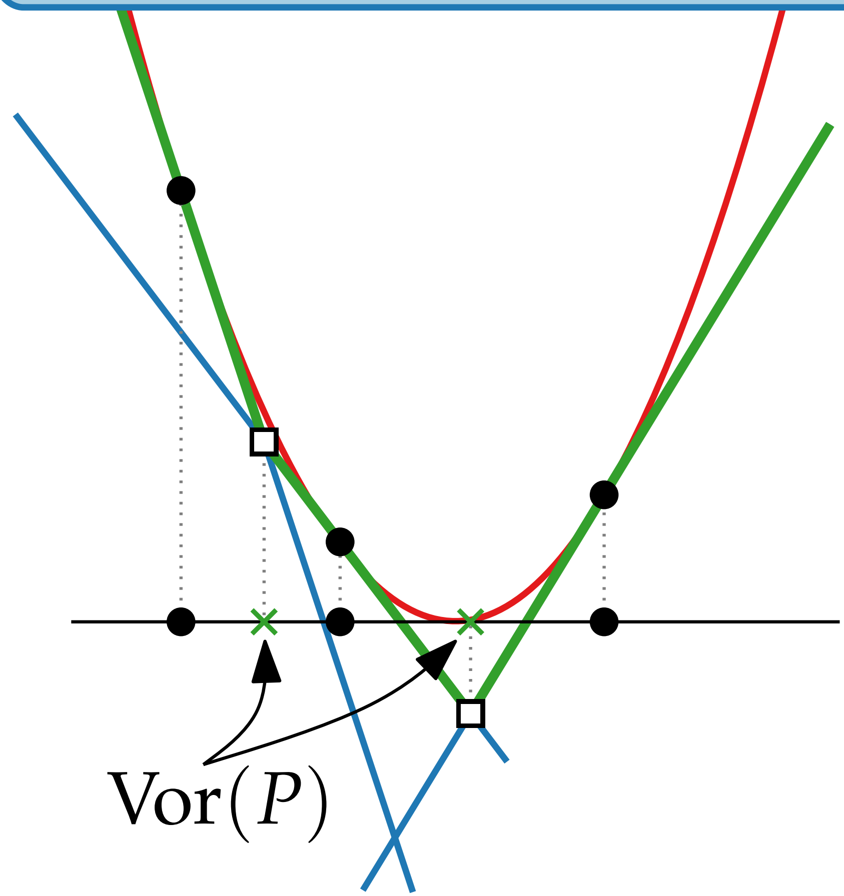


can compute $\mathrm{Vor}(P)$ in $\mathbb{R}^2$
via upper envelope in $\mathbb{R}^3$

upper envelope in $\mathbb{R}^3$ is in
one-to-one correspondence to
lower convex hull of pt set $\mathcal{H}^\star$

The Upper Envelope Strikes Back

Theorem. Let $P \subset \mathbb{R}^2 \times \{0\}$ and $\mathcal{H} = \{h(p) \mid p \in P\}$.
Let $\mathcal{E}(\mathcal{H})$ be the upper envelope of $\mathcal{H}$.
The projection of $\mathcal{E}(\mathcal{H})$ on $z = 0$ is $\mathrm{Vor}(P)$.



can compute $\mathrm{Vor}(P)$ in $\mathbb{R}^2$
via upper envelope in $\mathbb{R}^3$

upper envelope in $\mathbb{R}^3$ is in
one-to-one correspondence to
lower convex hull of pt set $\mathcal{H}^\star$

use algorithm Rand3DConvexHull!