13.1 Parametric sequence alignment

김창기

2016. 4. 26

목차

- **Introduction**
- **Definitions and first result**
- **Parametric alignment with the use of scoring matrices**
- **Efficient algorithms for computing a polygonal decomposition**
- **Time analysis and the next idea**
- **Bounding the number of polygons in the decomposition**
- **Uses for parametric alignment**

Introduction

- **When using sequence alignment method, there is often considerable disagreement about how to weight matches, mismatches, indels, and gaps.**

- ***Parametric sequence alignment* is a tool that efficiently explores such penalty variation.**
  - avoids the problem of choosing fixed parameter
  - computing the optimal alignment as a function of a variable parameters.

목차

Definitions and first result

- **Parametric alignment problems arise both with the use of character-specific scoring matrices and without their use.**

- **We first consider the case that no scoring matrix is used.**

Definitions and first result

- **Definition**
  - For any Alignment $\mathcal{A}$ of two strings, let $mt_{\mathcal{A}}$, $ms_{\mathcal{A}}$, $id_{\mathcal{A}}$, and $gp_{\mathcal{A}}$, respectively, denote the number of matches, mismatches, indels, and gaps contained in $\mathcal{A}$.
    without the use of character-specific scoring matrices, the value of $\mathcal{A}$ is

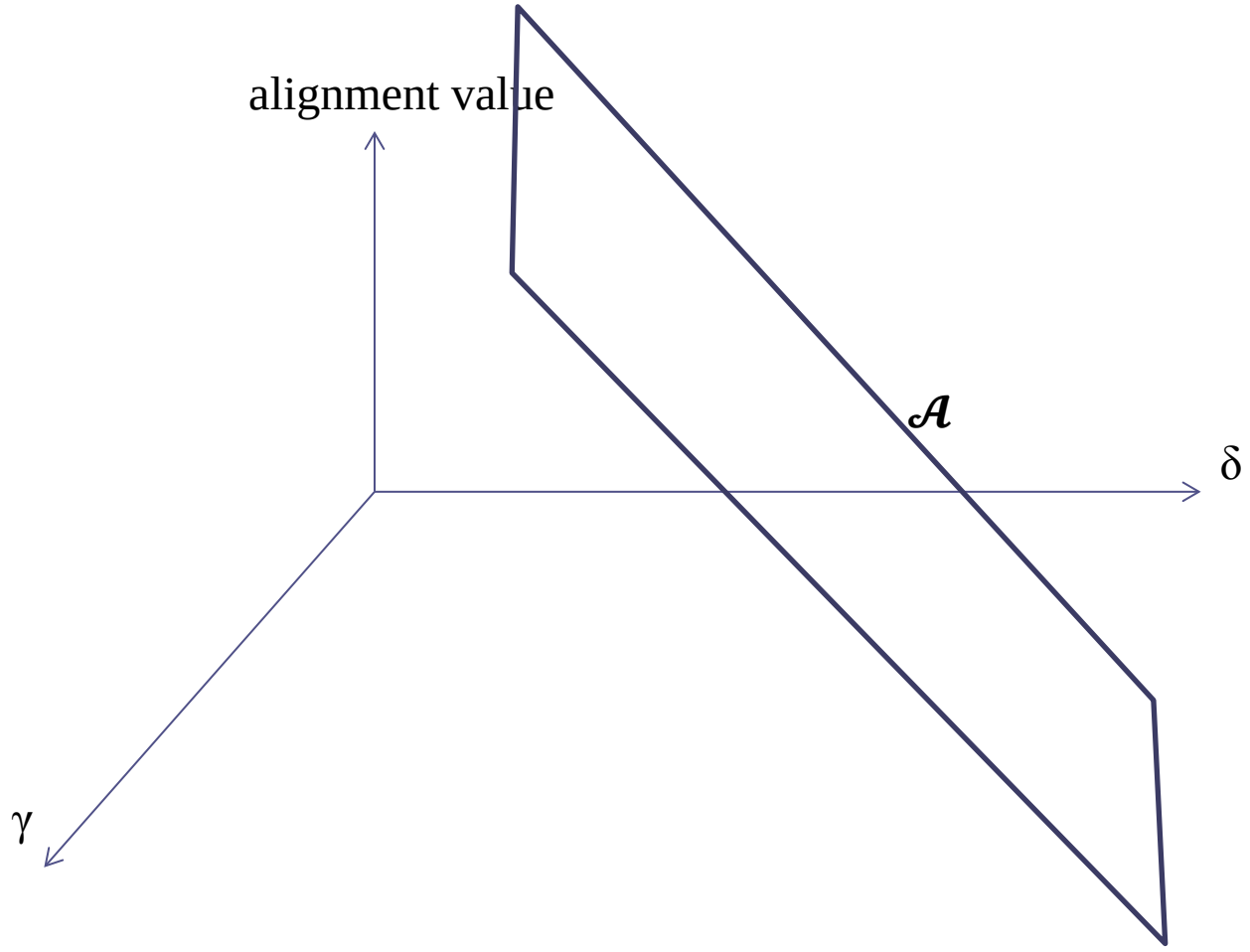
$$v_{\mathcal{A}}(\alpha, \beta, \gamma, \delta) \equiv \alpha * mt_{\mathcal{A}} - \beta * ms_{\mathcal{A}} - \gamma * id_{\mathcal{A}} - \delta * gp_{\mathcal{A}}$$

- **Where α, β, γ, and δ are parameters that can be varied to adjust the relative contributions of the matches, mismatches, indels, and gaps.**

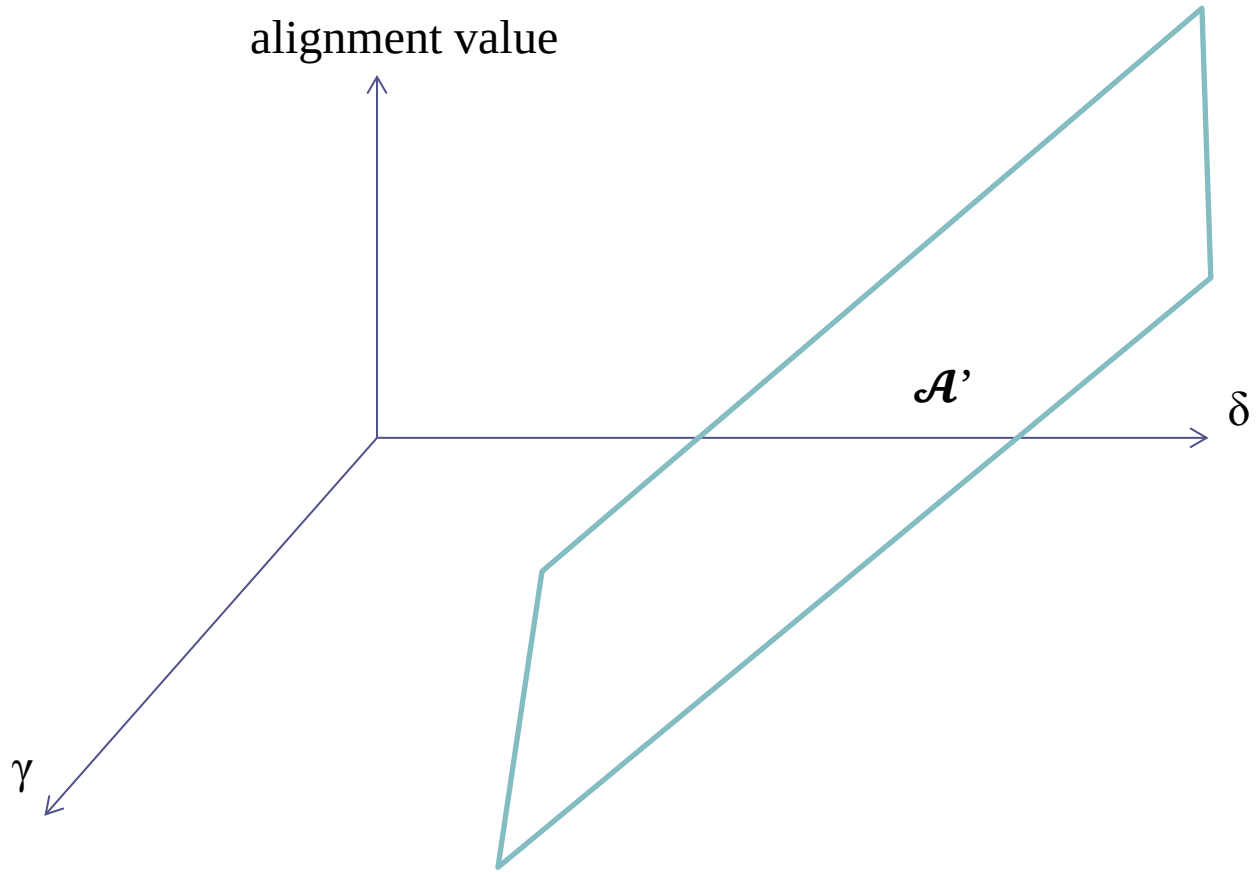
Definitions and first result

- $v_{\mathcal{A}}(\alpha, \beta, \gamma, \delta) \equiv \alpha * mt_{\mathcal{A}} - \beta * ms_{\mathcal{A}} - \gamma * id_{\mathcal{A}} - \delta * gp_{\mathcal{A}}$
  - If four parameters have fixed values α_0, β_0, γ_0, δ_0, resulting problem is the standard fixed-parameter problem.
  - We will restrict attention to parametric problems where only two of the four parameters(γ, δ) are varied for illustration. (The remaining parameters are given fixed values.)

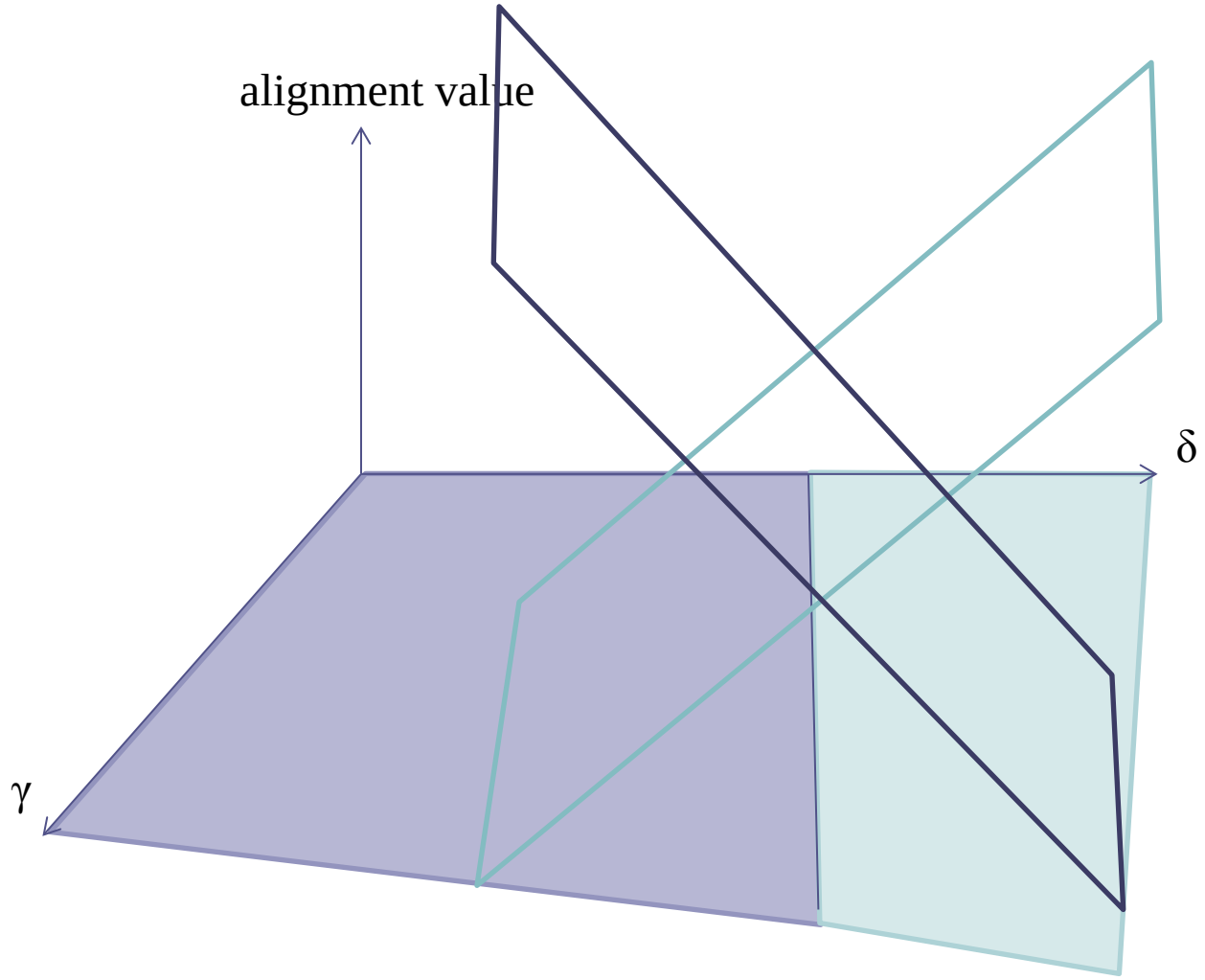
Definitions and first result



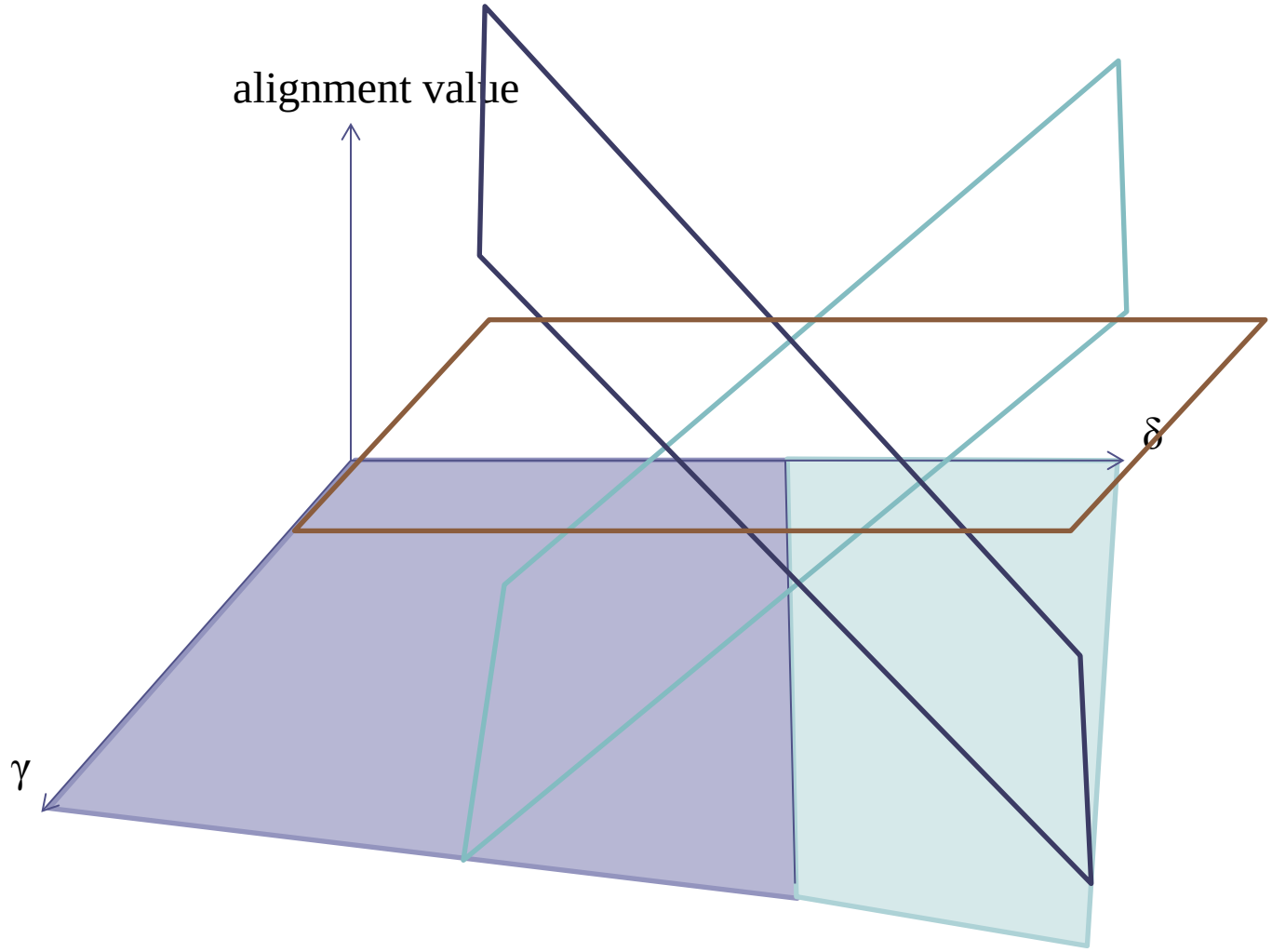
Definitions and first result



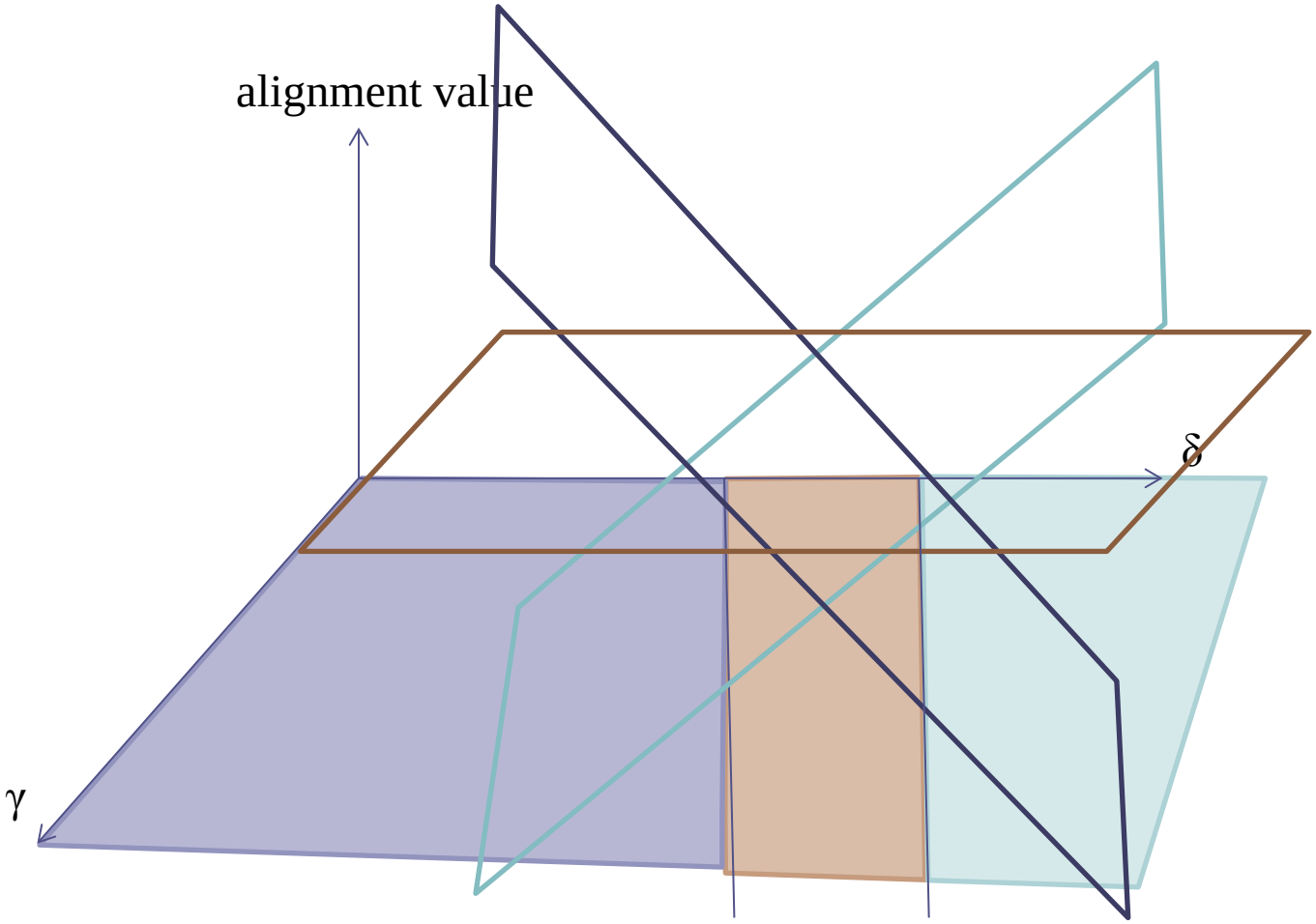
Definitions and first result



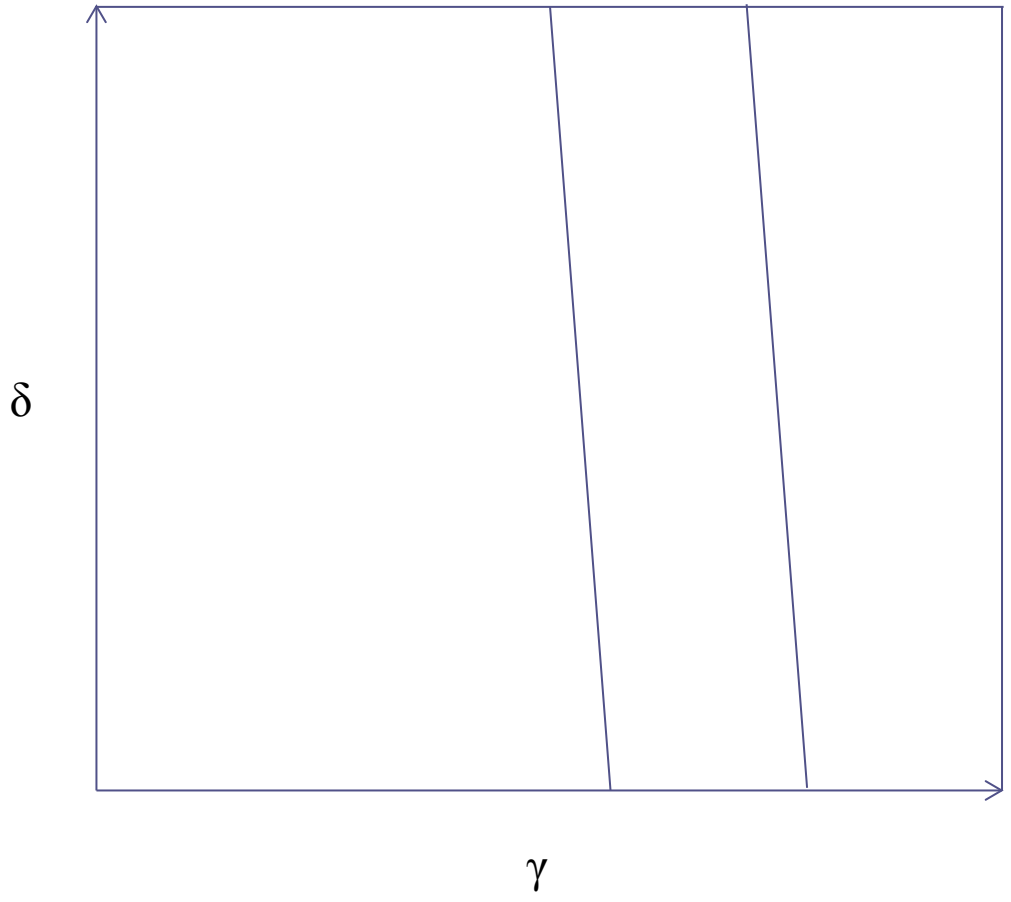
Definitions and first result



Definitions and first result



Definitions and first result



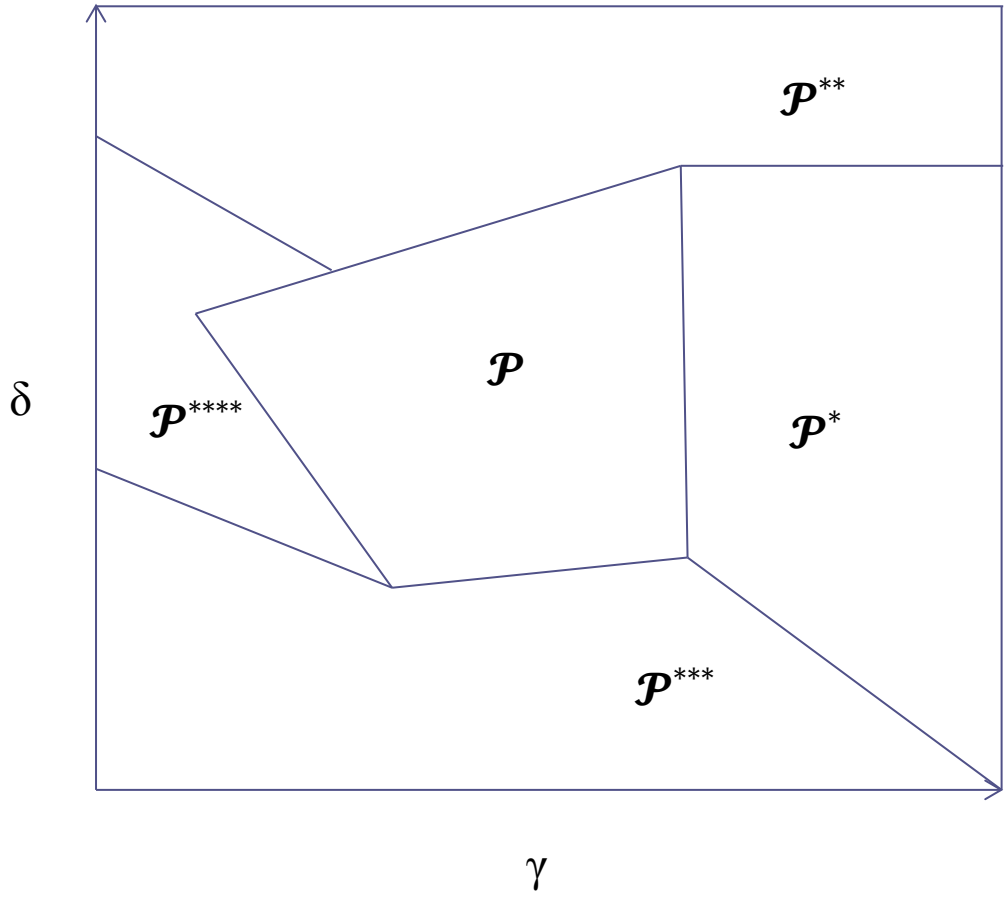
Definitions and first result

- **Lemma 13.1.1**
  - If the planes for alignments $\mathcal{A}$ and $\mathcal{A}$' intersect and are distinct, then there is a line L in the γ, δ space along which $\mathcal{A}$ and $\mathcal{A}$' have equal value; $\mathcal{A}$ has larger value than $\mathcal{A}$' on one of the half-planes defined by L, and $\mathcal{A}$ has smaller value on the other half-plane. If the planes for $\mathcal{A}$ and $\mathcal{A}$' do not intersect, then one of the alignments has larger value than the other at every γ, δ point.

Definitions and first result

- **Corollary 13.1.1**
  - If $\mathcal{A}$ is optimal for at least one point p in the γ, δ space, then it is optimal only for point p, or it is optimal only for a line segment that contains p, or it is optimal only for a convex polygon that contains p.
- **Then it follows that**
- **Theorem 13.1.1**
  - Given two strings S_1 and S_2, the γ, δ parameter space decomposes into *convex polygons* such that any alignment that is optimal for some γ, δ point in the interior of a polygon $\mathcal{P}$ is optimal for all points in $\mathcal{P}$ and nowhere else.

Definitions and first result



목차

- Introduction
- Definitions and first result
- **Parametric alignment with the use of scoring matrices**
- **Efficient algorithms for computing a polygonal decomposition**
- **Time analysis and the next idea**
- **Bounding the number of polygons in the decomposition**
- **Uses for parametric alignment**

Parametric alignment with the use of scoring matrices

- **Definition**
  - For any alignment $\mathcal{A}$ of two strings, let $smt_{\mathcal{A}}$ and $sms_{\mathcal{A}}$, respectively, denote the total score (obtained from the scoring matrix) for the specific matches in $\mathcal{A}$ and the total score for the specific mismatches in $\mathcal{A}$. As before, $id_{\mathcal{A}}$ and $gp_{\mathcal{A}}$ denote the number of indels and gaps contained in $\mathcal{A}$

- **Using scoring matrices, the parametric value of alignment $\mathcal{A}$ is**

$$\alpha * smt_{\mathcal{A}} + \beta * sms_{\mathcal{A}} - \gamma * id_{\mathcal{A}} - \delta * gp_{\mathcal{A}}$$

목차

- Introduction
- Definitions and first result
- Parametric alignment with the use of scoring matrices
- **Efficient algorithms for computing a polygonal decomposition**
- **Time analysis and the next idea**
- **Bounding the number of polygons in the decomposition**
- **Uses for parametric alignment**

Efficient algorithms for computing a polygonal decomposition

- **Ray-search problem**
  - Given an alignment $\mathcal{A}$, a point p where is optimal, and a ray h in γ, δ space starting at p, find the furthest point (call it r^*) from p on ray h where remains $\mathcal{A}$ optimal. If $\mathcal{A}$ remains optimal until h reaches a border of the parameter space, then r^* is that border point on h. it is also possible that $r^* = p$.

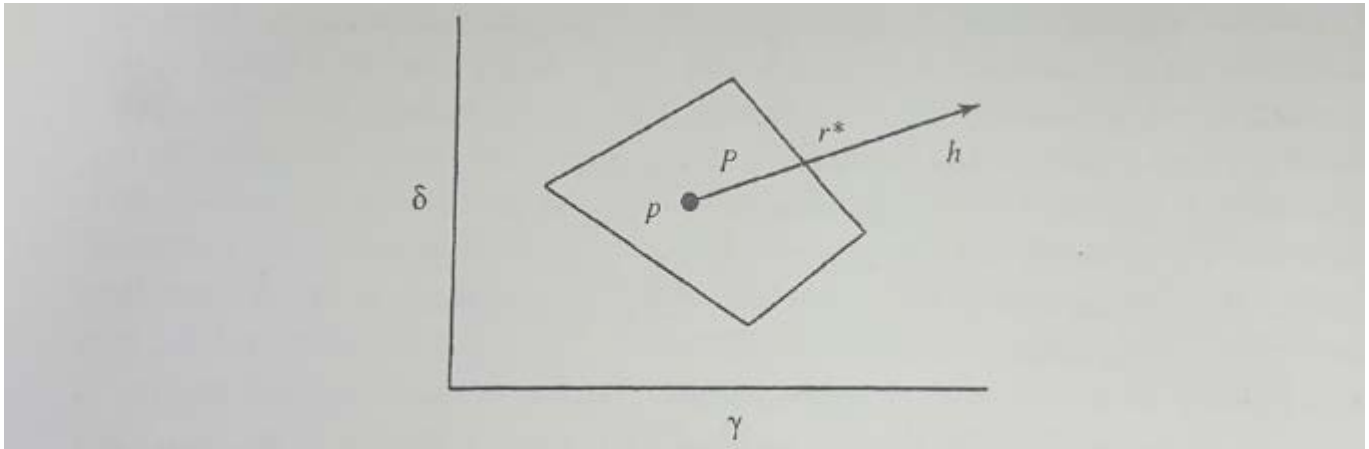


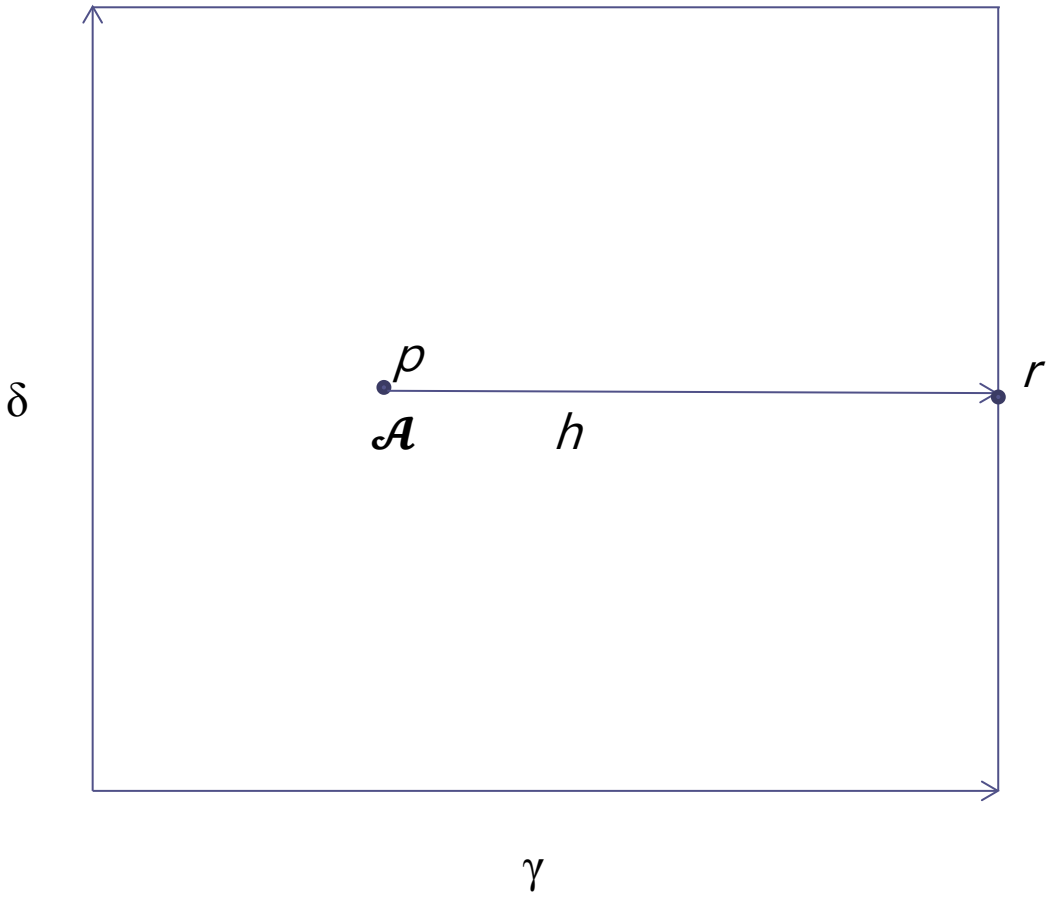
Figure 13.2: Alignment $\mathcal{A}$ is optimal in polygon P. Starting from point p, $\mathcal{A}$ remains optimal along ray h through point r^*.

Efficient algorithms for computing a polygonal decomposition

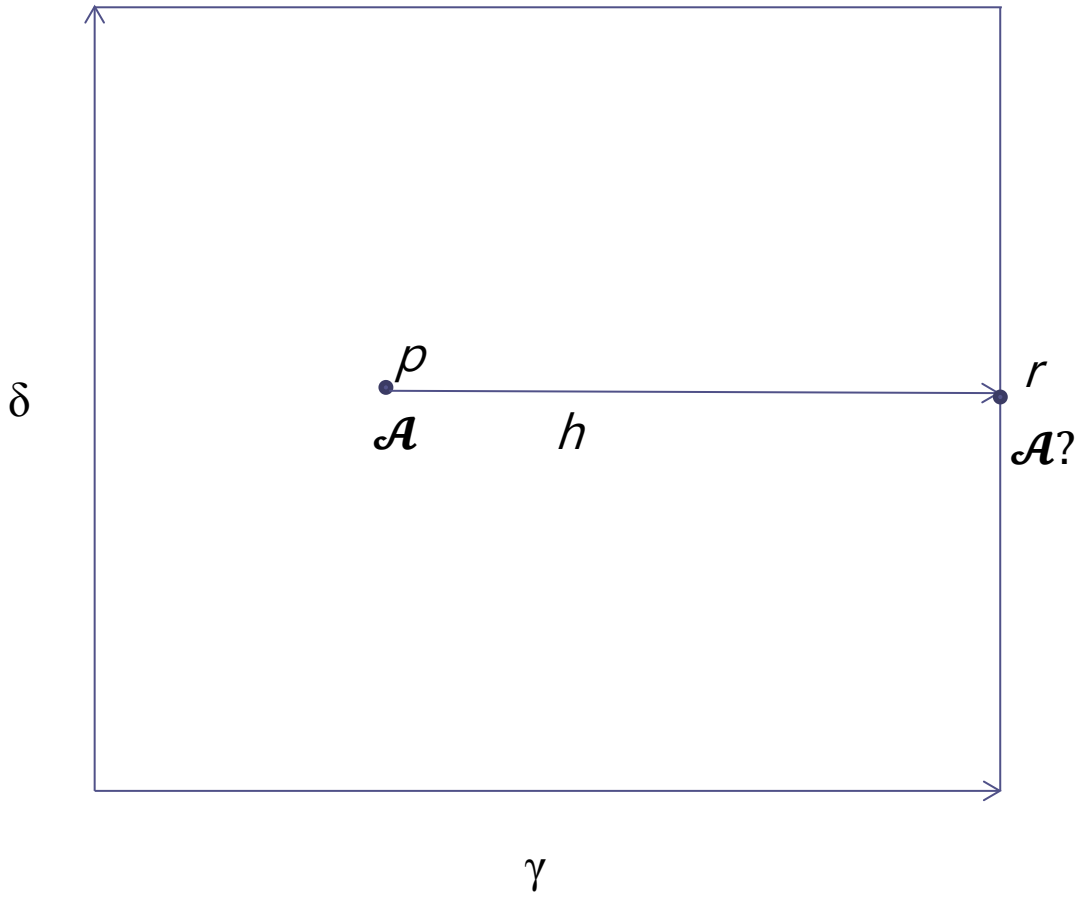
- **Newton's ray-search algorithm**
  - Set r to the (γ, δ) point where h intersects a border of the parameter space
  - While $\mathcal{A}$ is not an optimal alignment at point r do
    begin
      Find an optimal alignment $\mathcal{A}^*$ at point r
      Set r to be the unique point on h where the value of $\mathcal{A}$ equals the value of $\mathcal{A}^*$
    end
  - Set r^* to r

- **The following three facts will be needed in the analysis.**

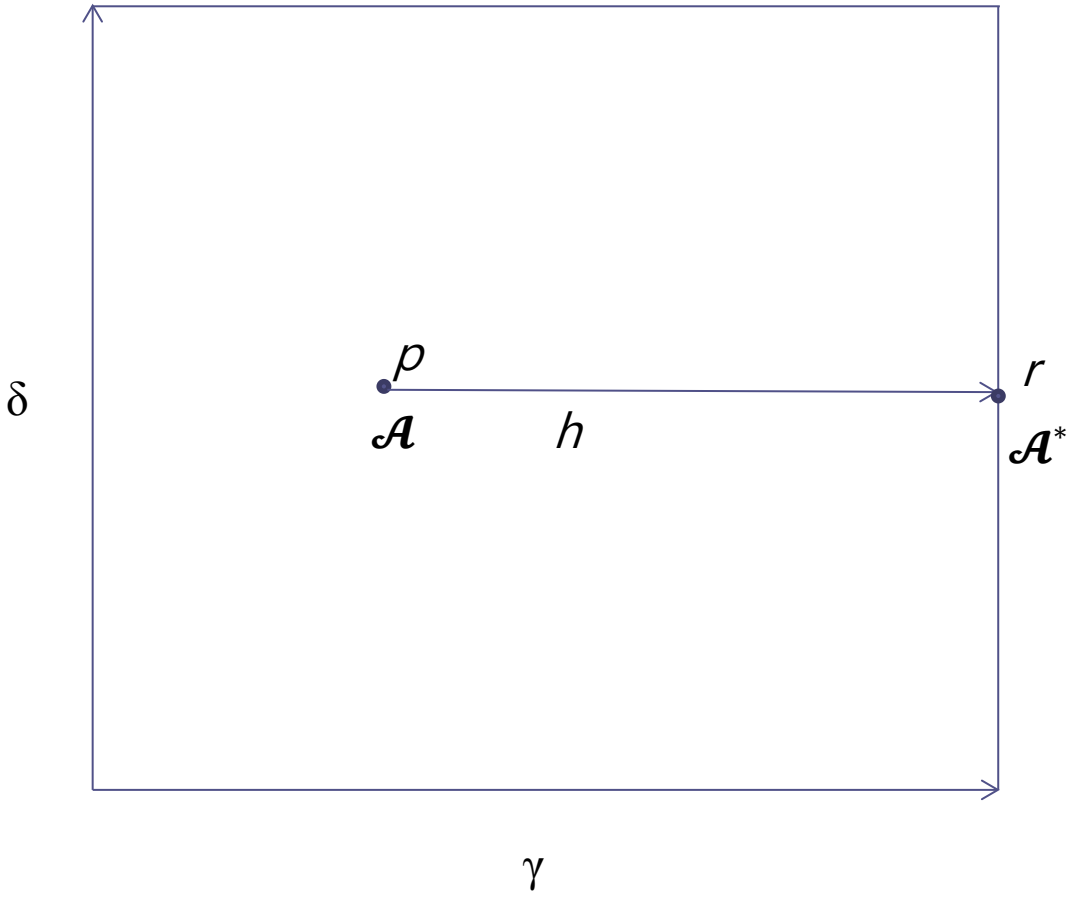
Efficient algorithms for computing a polygonal decomposition



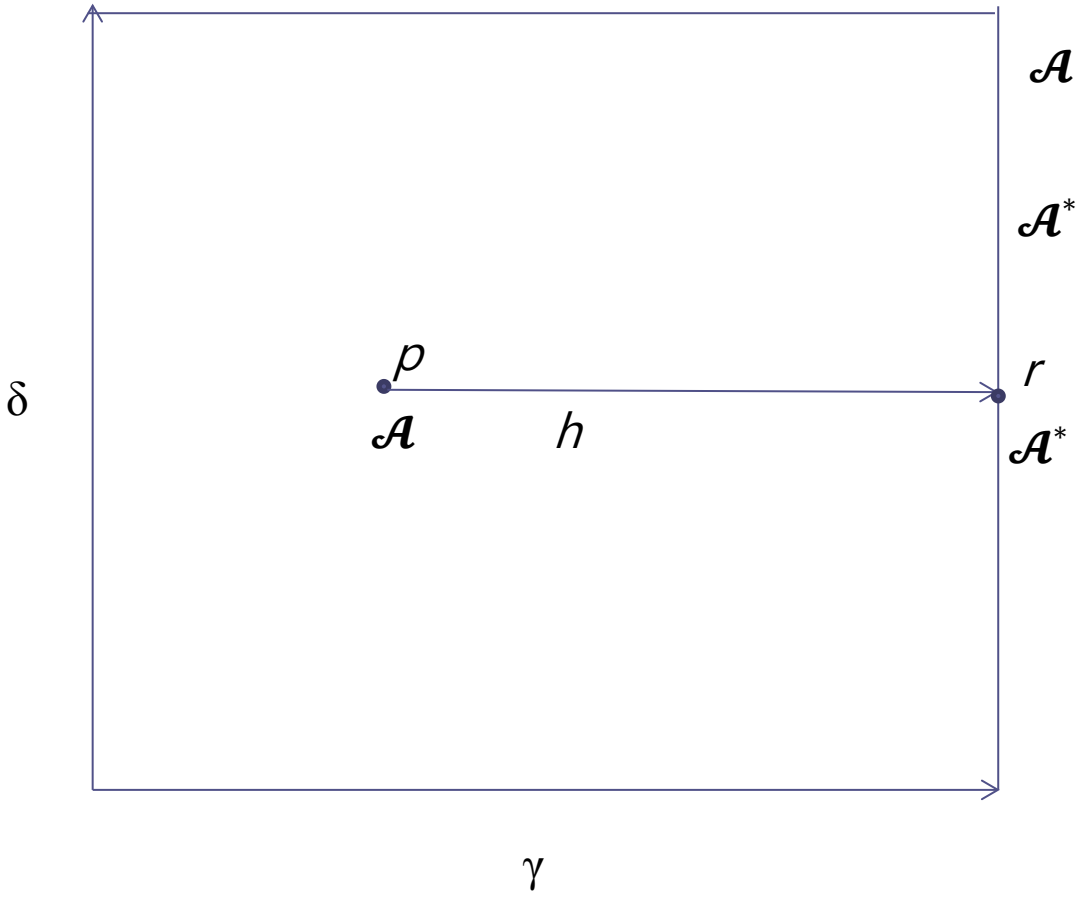
Efficient algorithms for computing a polygonal decomposition



Efficient algorithms for computing a polygonal decomposition



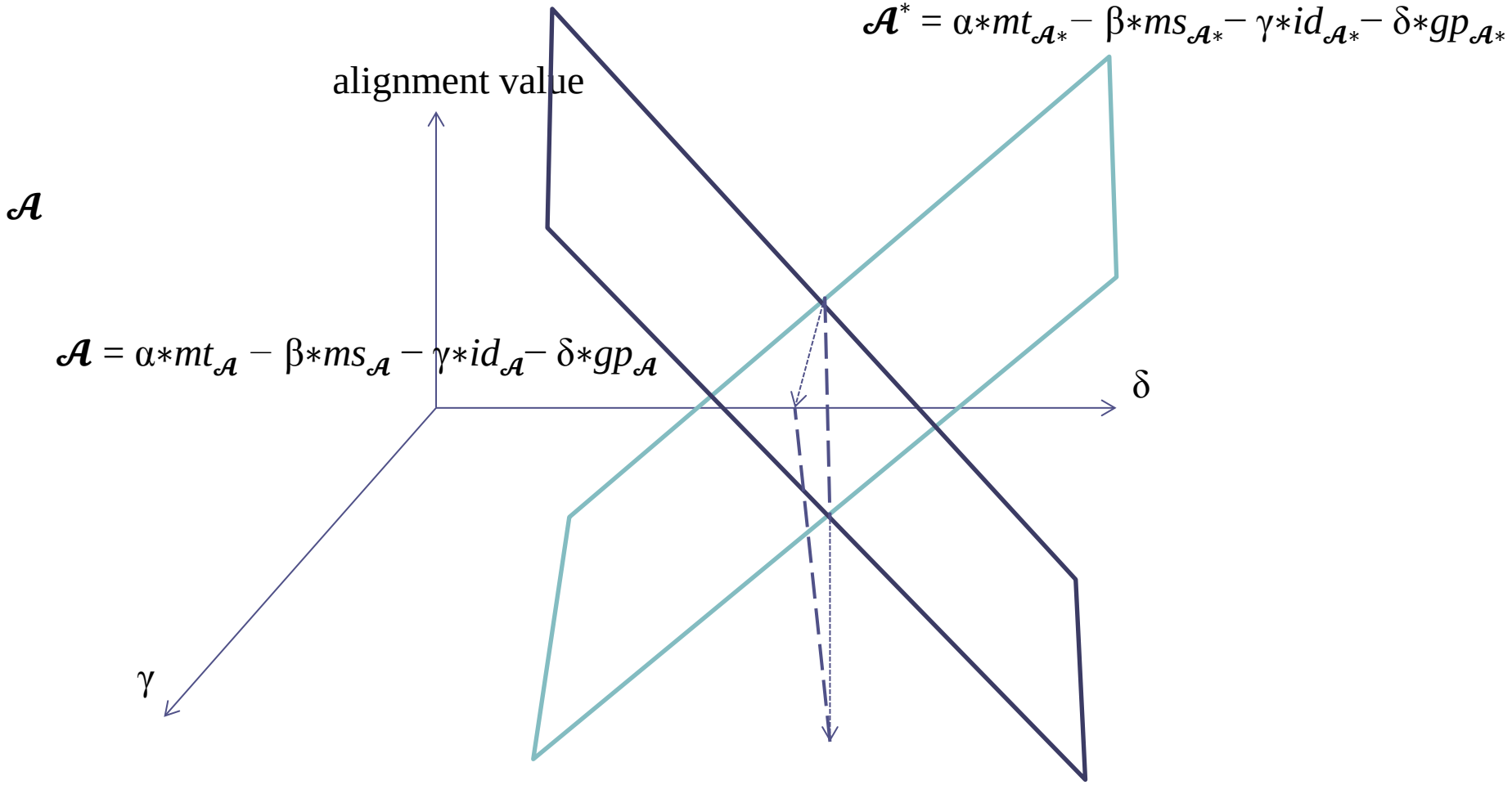
Efficient algorithms for computing a polygonal decomposition



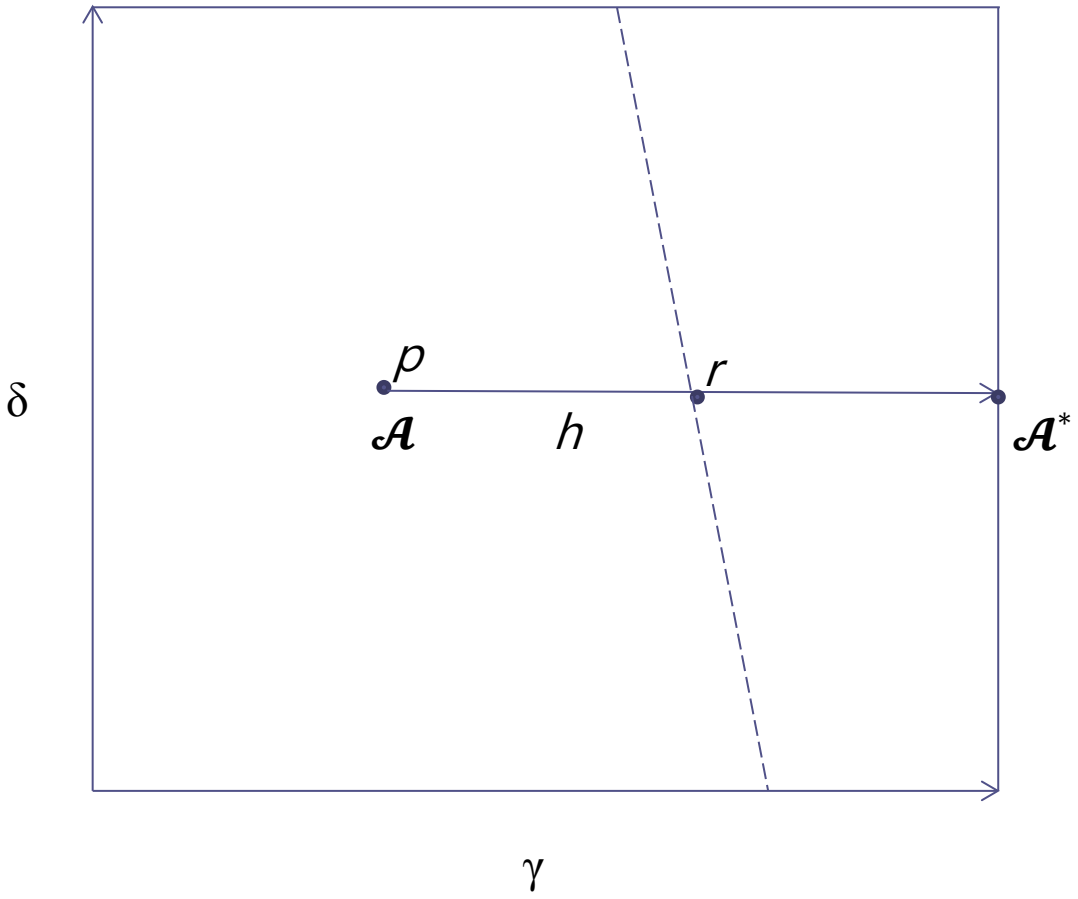
$$\mathcal{A} = \alpha * mt_{\mathcal{A}} - \beta * ms_{\mathcal{A}} - \gamma * id_{\mathcal{A}} - \delta * gp_{\mathcal{A}}$$

$$\mathcal{A}^* = \alpha * mt_{\mathcal{A}^*} - \beta * ms_{\mathcal{A}^*} - \gamma * id_{\mathcal{A}^*} - \delta * gp_{\mathcal{A}^*}$$

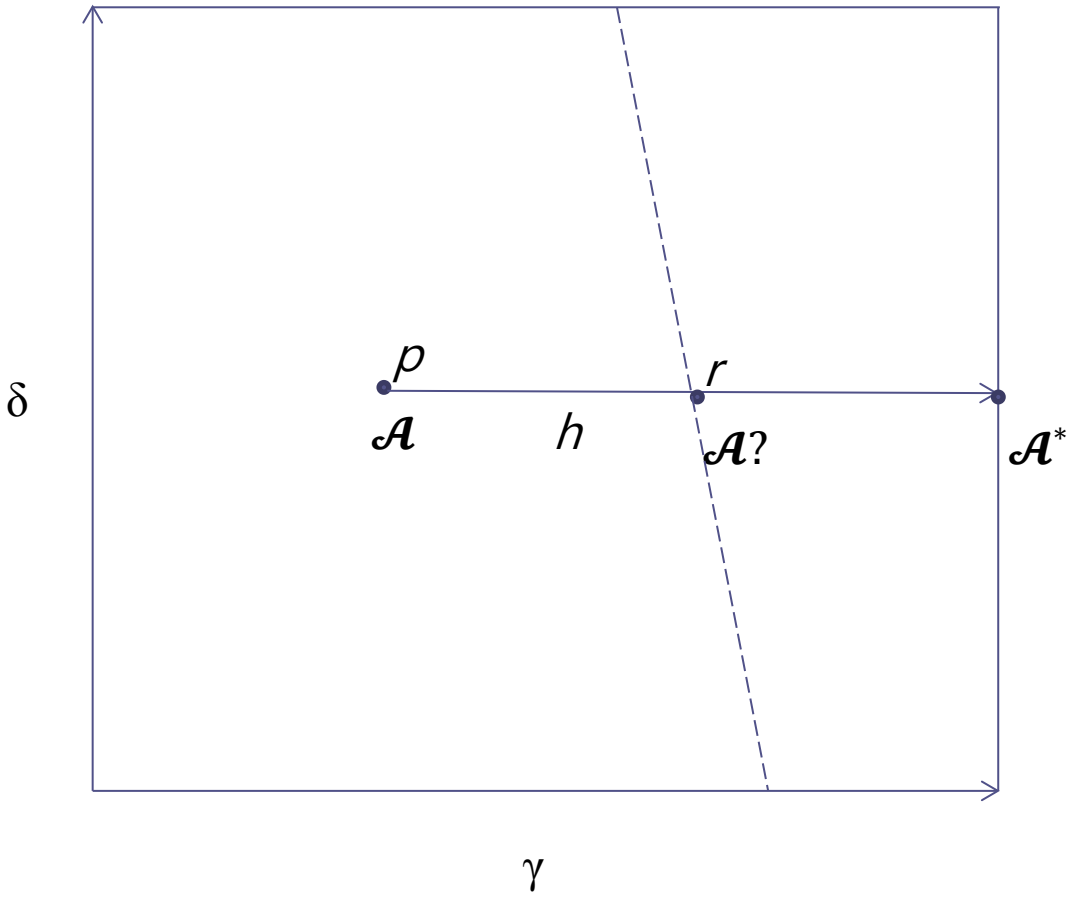
Efficient algorithms for computing a polygonal decomposition



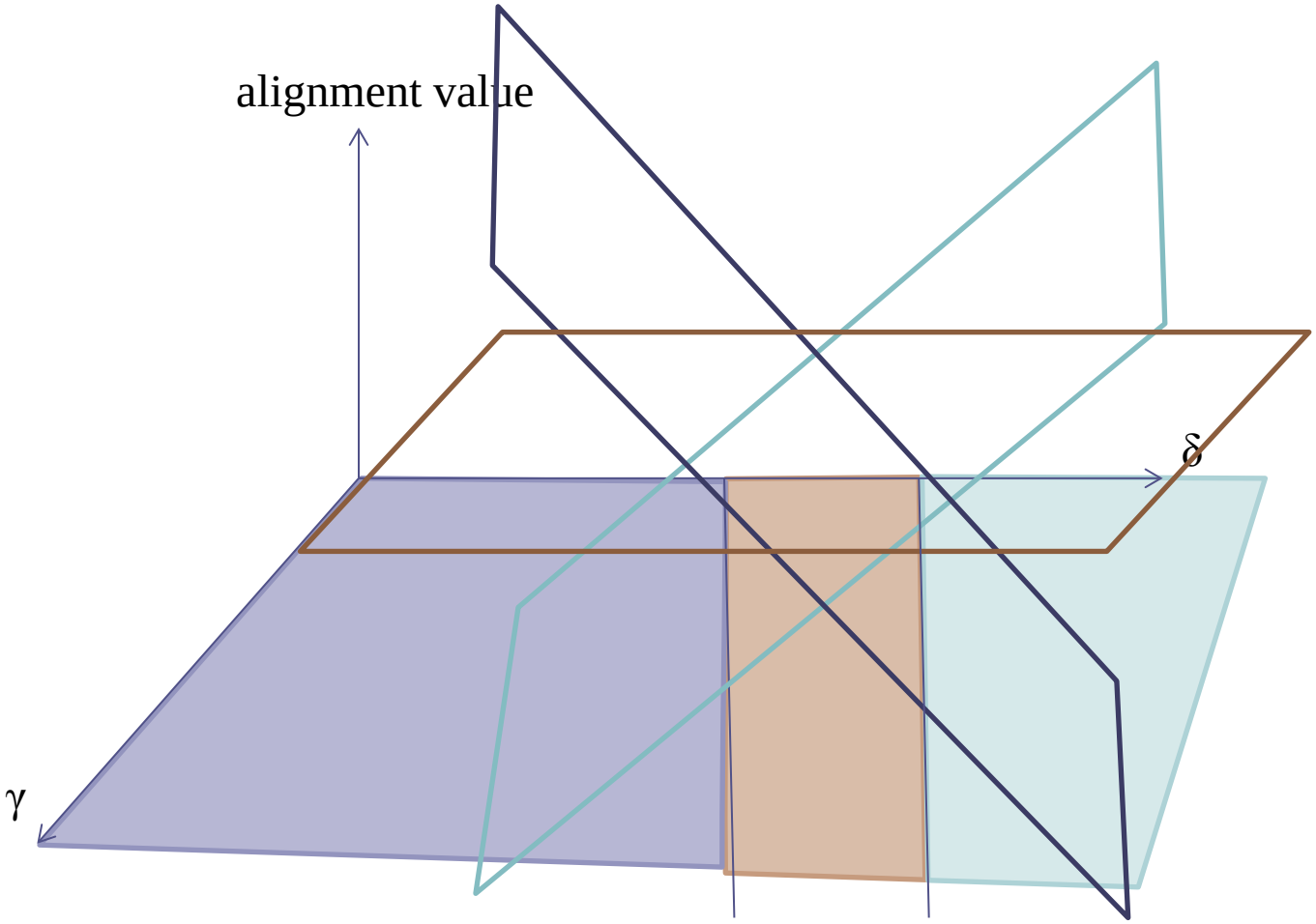
Efficient algorithms for computing a polygonal decomposition



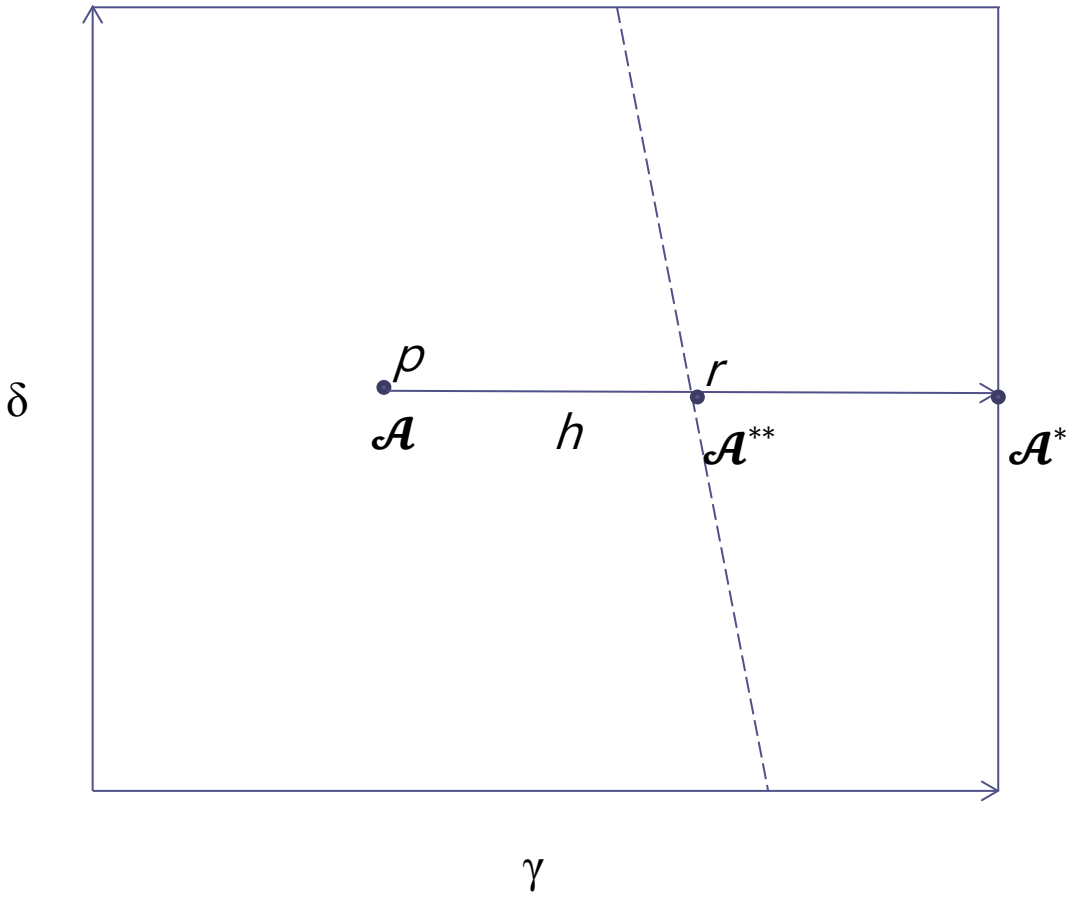
Efficient algorithms for computing a polygonal decomposition



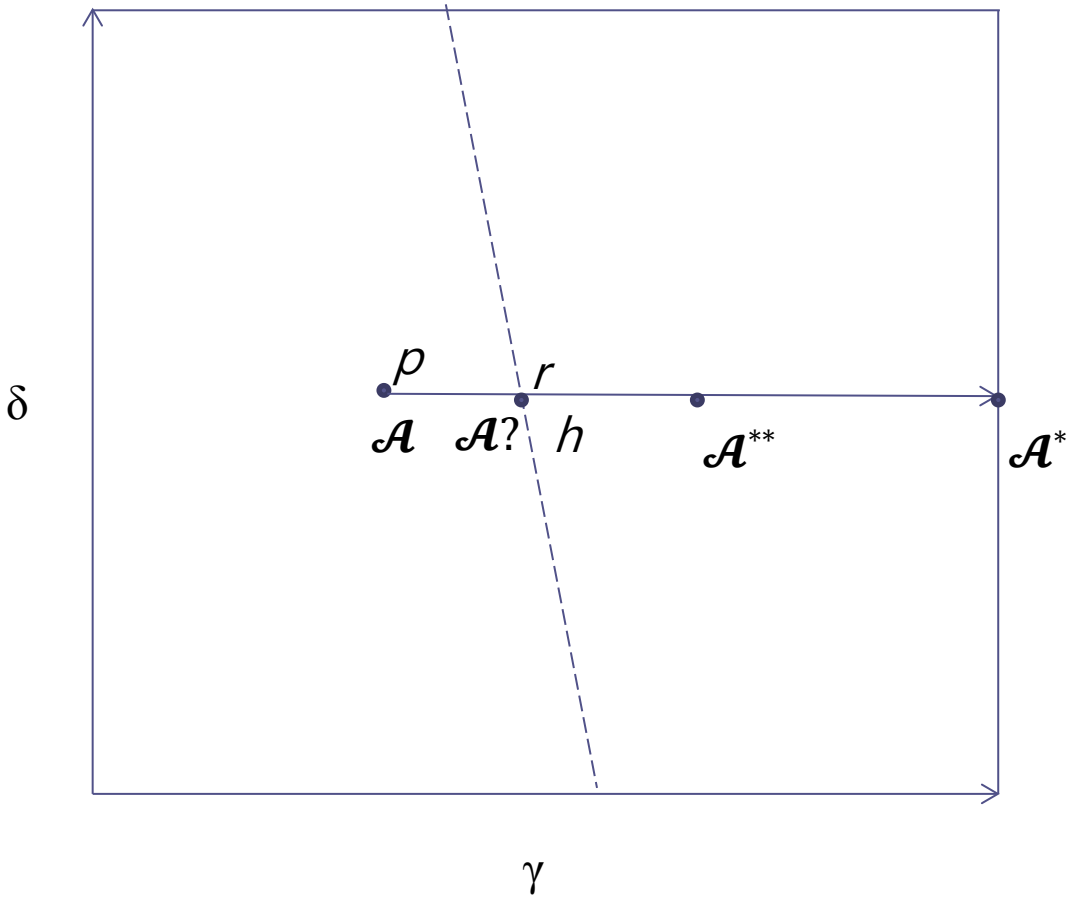
Definitions and first result



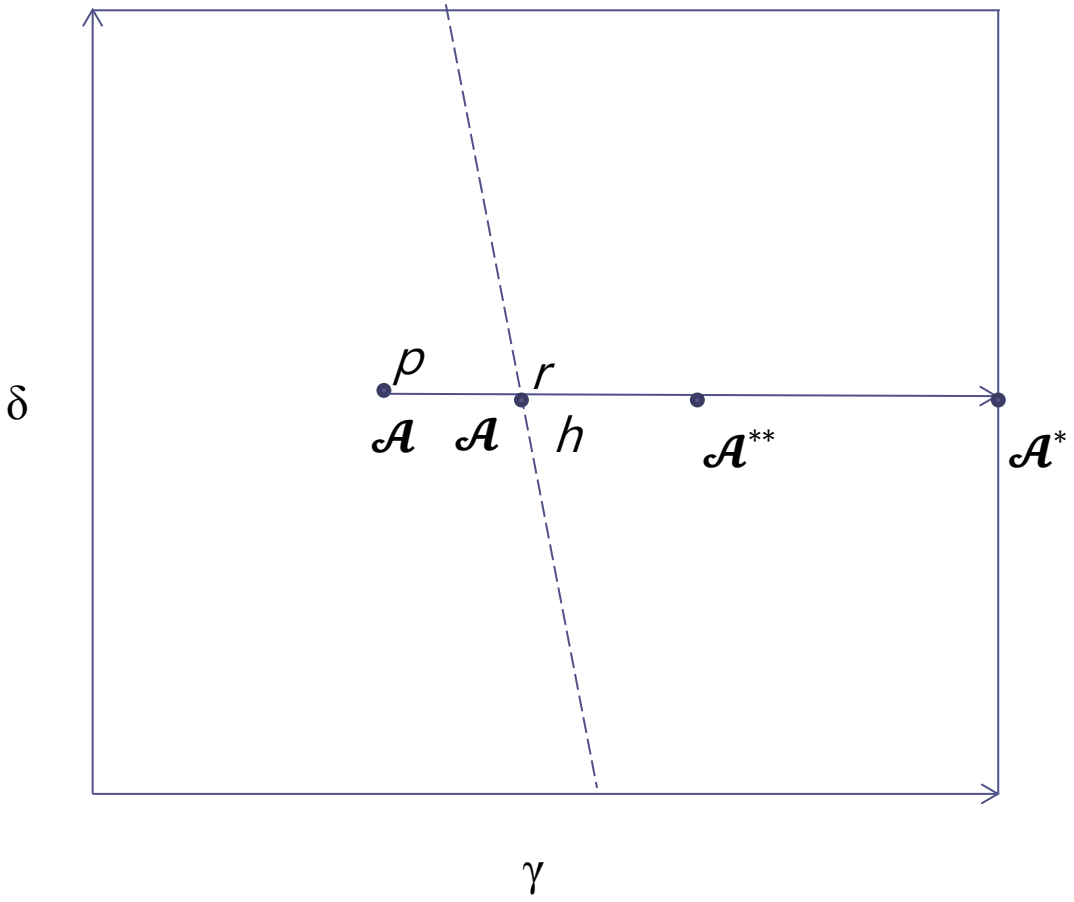
Efficient algorithms for computing a polygonal decomposition



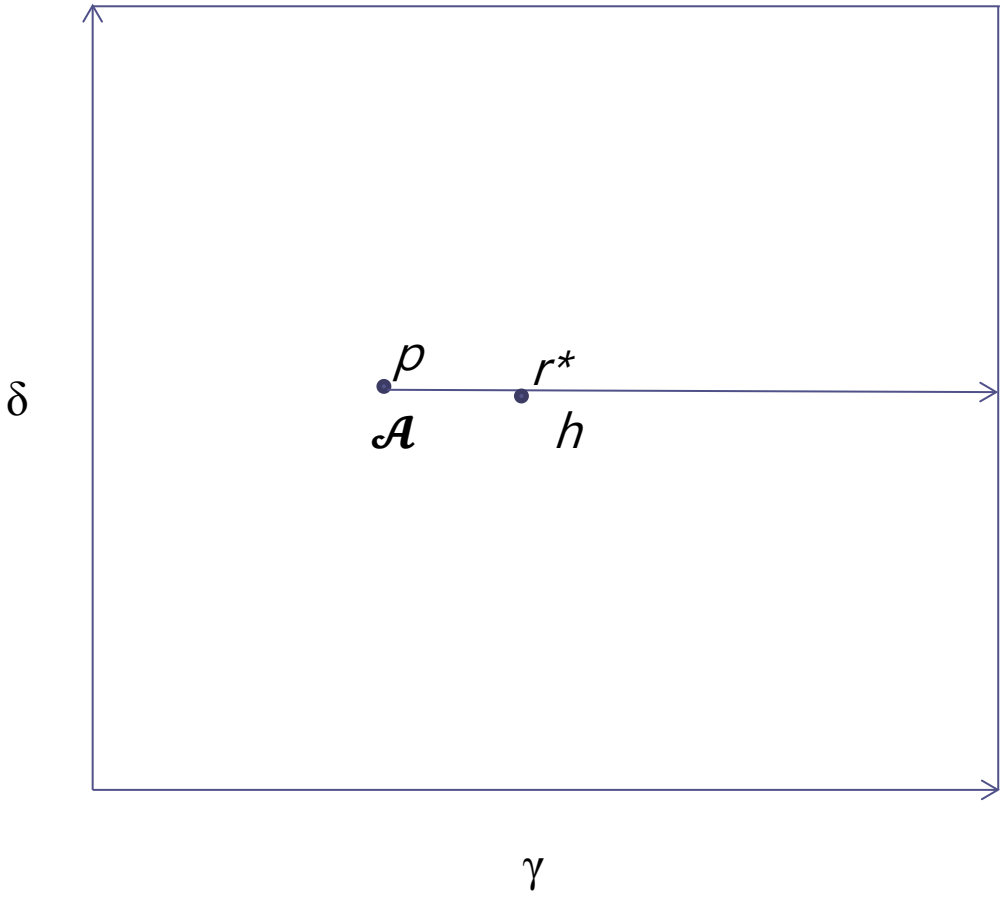
Efficient algorithms for computing a polygonal decomposition



Efficient algorithms for computing a polygonal decomposition



Efficient algorithms for computing a polygonal decomposition



Efficient algorithms for computing a polygonal decomposition

- **Newton's ray-search algorithm**
  - Set r to the (γ, δ) point where h intersects a border of the parameter space
  - While $\mathcal{A}$ is not an optimal alignment at point r do
    begin
      Find an optimal alignment $\mathcal{A}^*$ at point r
      Set r to be the unique point on h where the value of $\mathcal{A}$ equals the value of $\mathcal{A}^*$
    end
  - Set r^* to r

- **The following three facts will be needed in the analysis.**

Efficient algorithms for computing a polygonal decomposition

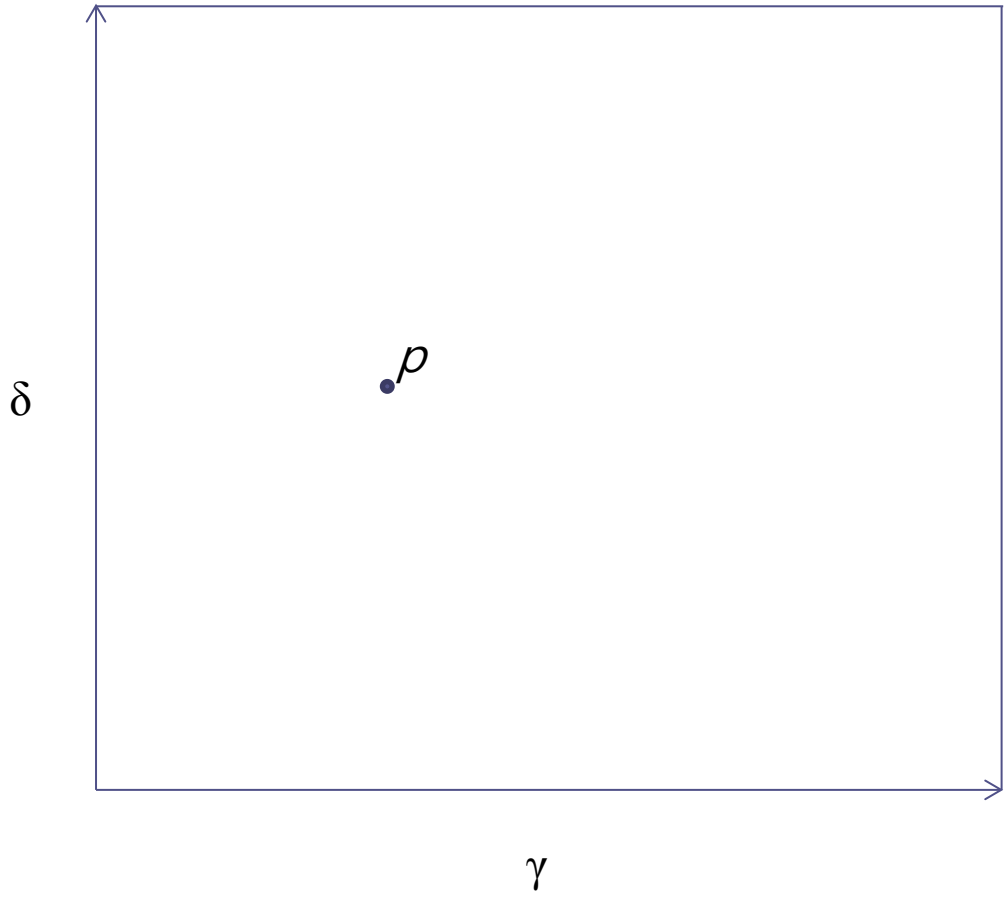
- **Lemma 13.1.2**
  - 1) Newton's ray-search algorithm finds r^* exactly.
  - 2) Unless $\mathcal{A}$ is optimal at the initial setting of r, the last computed alignment $\mathcal{A}^*$ is co-optimal with $\mathcal{A}$ at r^* and yet is also optimal on h for some nonzero distance beyond r^*
  - 3) When Newton's ray-search algorithm computes an alignment at a point r on h, none of the alignments computed previously(in this execution of Newton's algorithm) are optimal at r

Efficient algorithms for computing a polygonal decomposition

- **Finding a polygon of the decomposition**
  - Let $\mathcal{A}$ be an alignment that is optimal in the interior of an unknown polygon $\mathcal{P}(\mathcal{A})$
  - Let p be a known point where $\mathcal{A}$ is optimal.

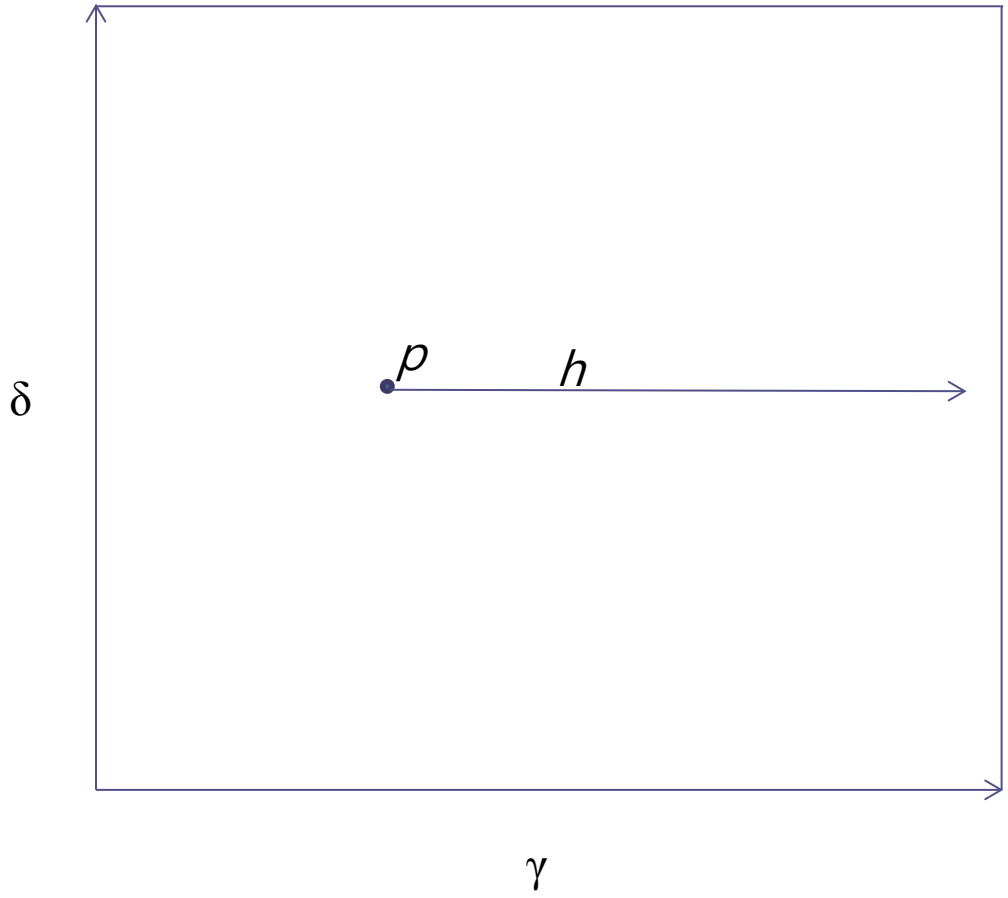
Efficient algorithms for computing a polygonal decomposition

- **Finding a polygon of the decomposition**



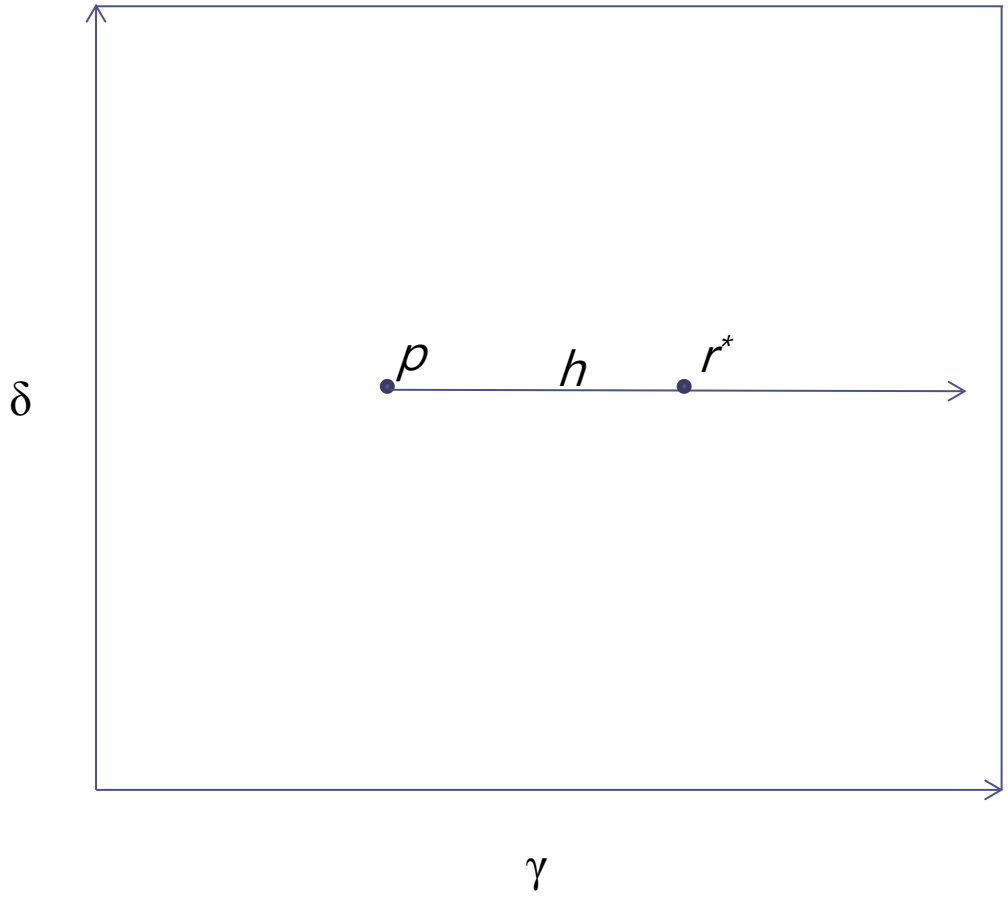
Efficient algorithms for computing a polygonal decomposition

- **Finding a polygon of the decomposition**



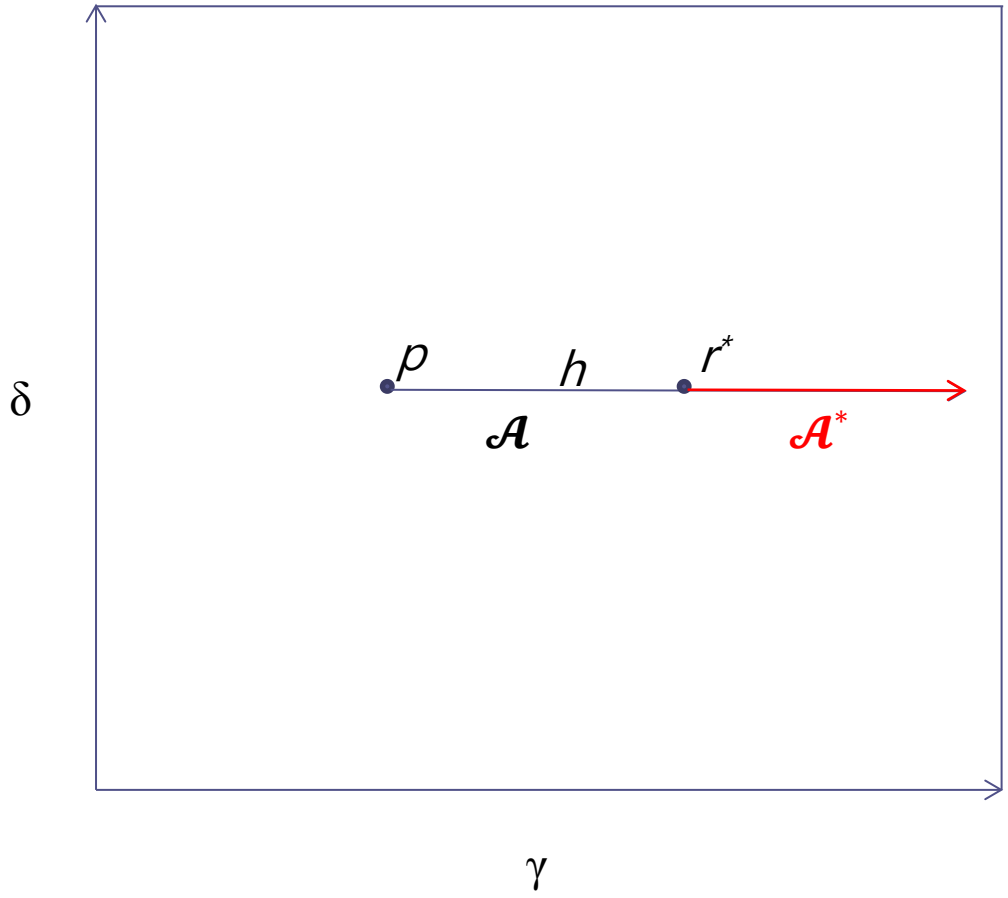
Efficient algorithms for computing a polygonal decomposition

- **Finding a polygon of the decomposition**



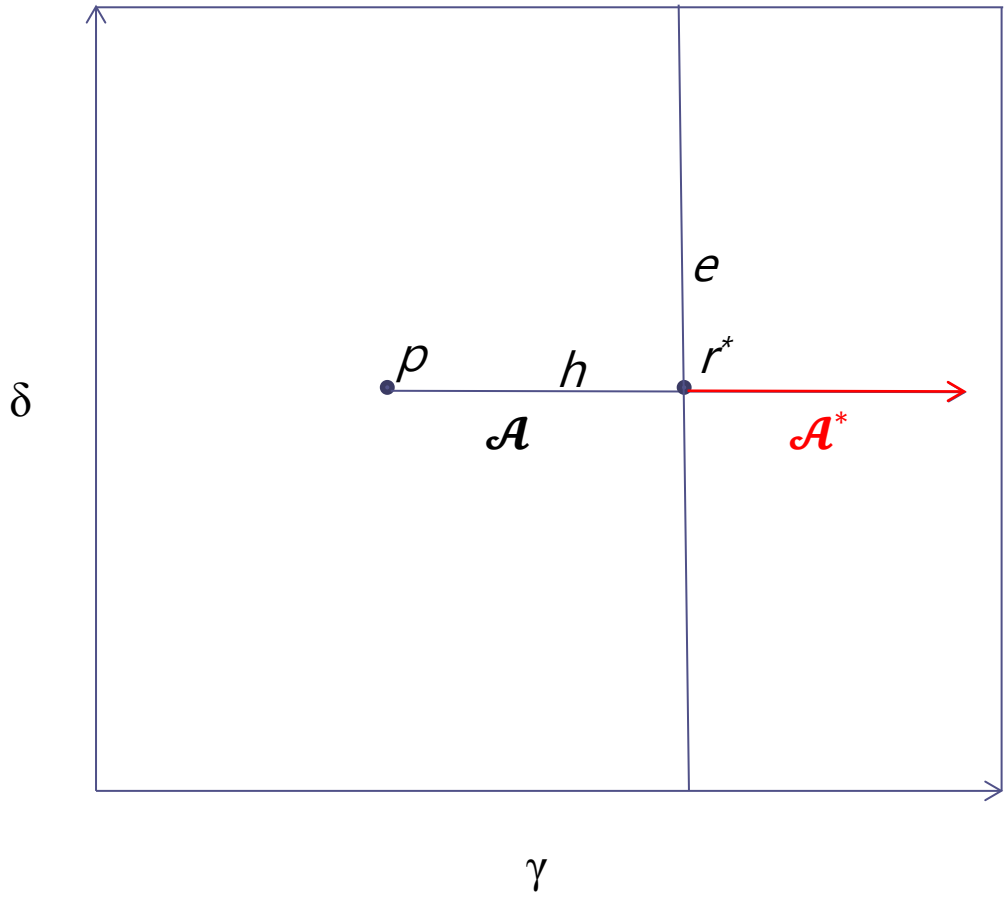
Efficient algorithms for computing a polygonal decomposition

- **Finding a polygon of the decomposition**



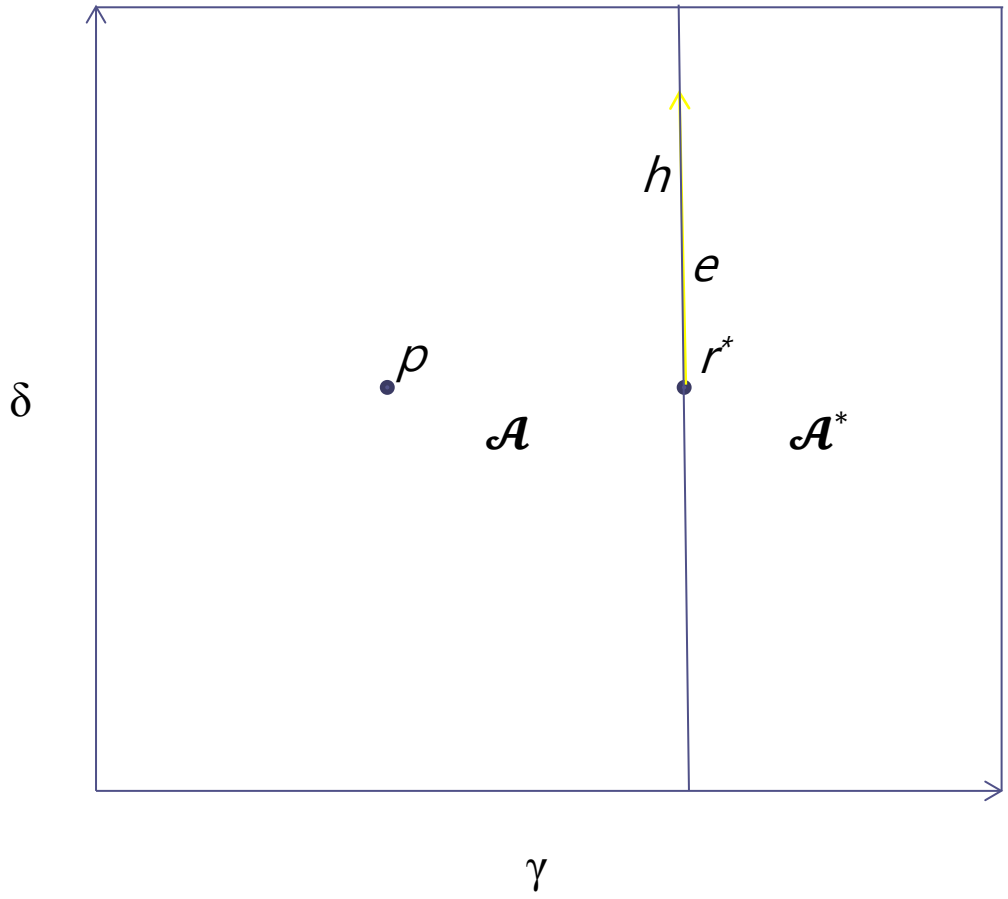
Efficient algorithms for computing a polygonal decomposition

- **Finding a polygon of the decomposition**



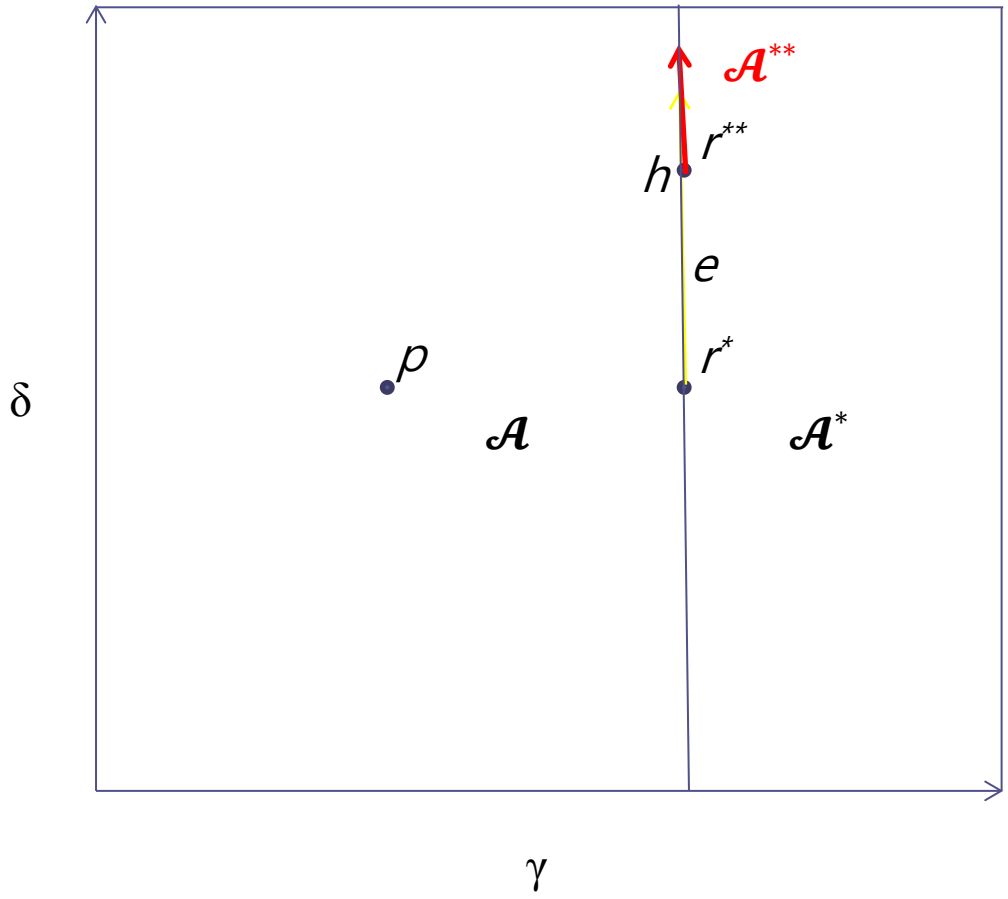
Efficient algorithms for computing a polygonal decomposition

- **Finding a polygon of the decomposition**



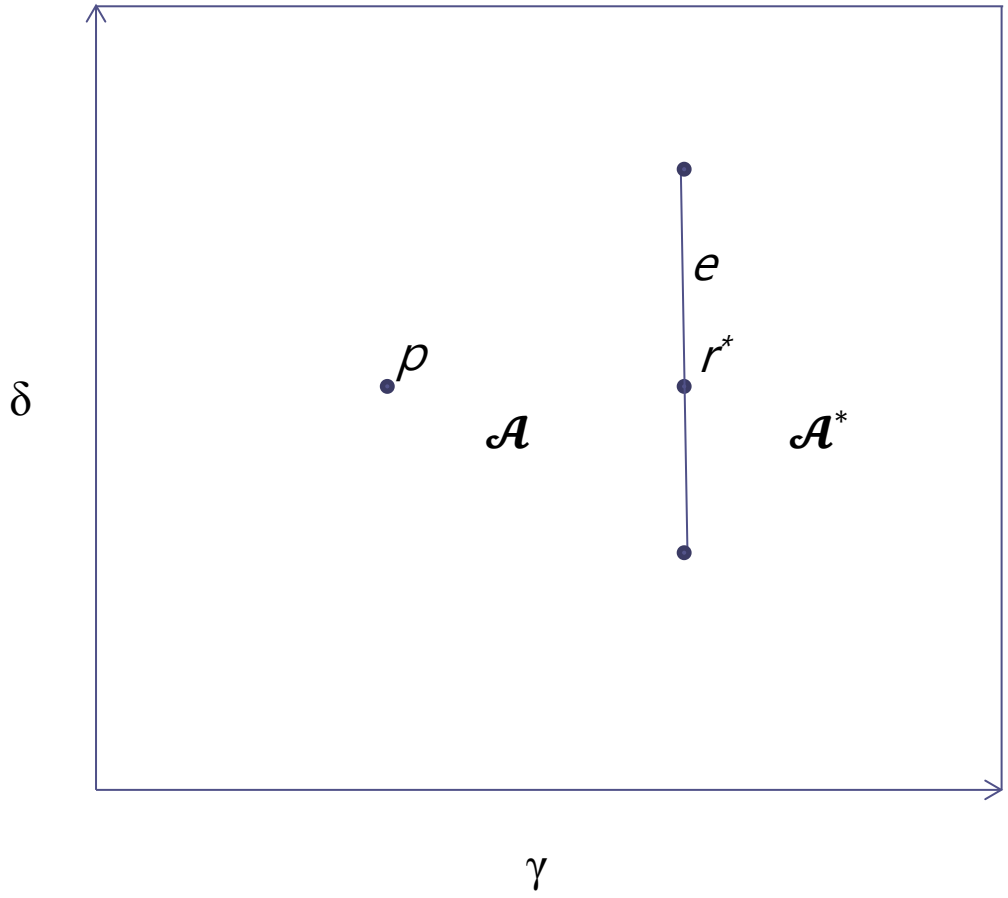
Efficient algorithms for computing a polygonal decomposition

- **Finding a polygon of the decomposition**



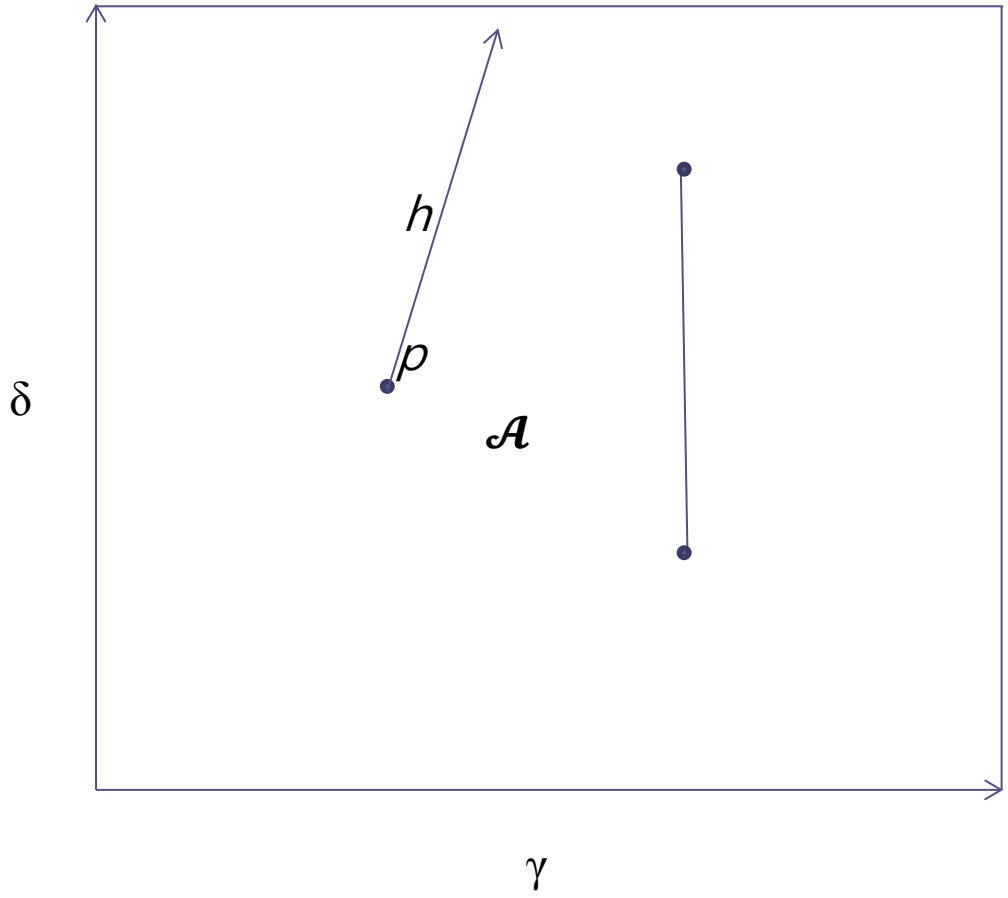
Efficient algorithms for computing a polygonal decomposition

- **Finding a polygon of the decomposition**



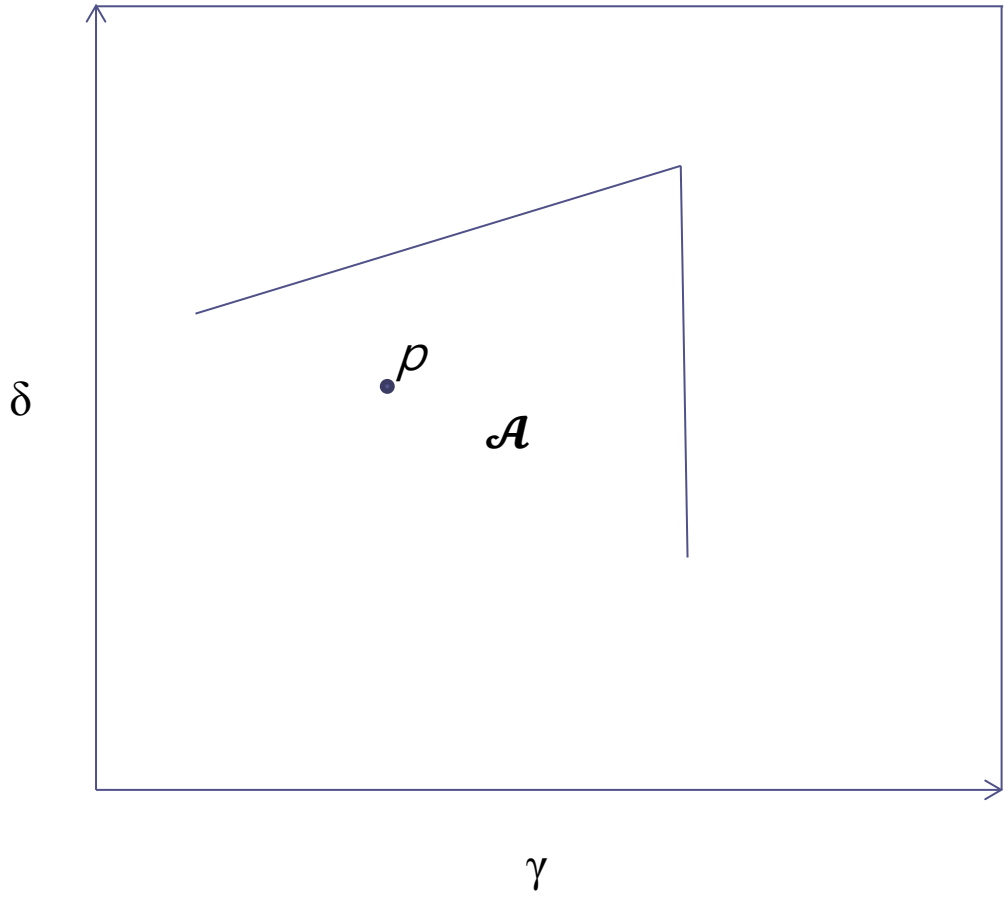
Efficient algorithms for computing a polygonal decomposition

- **Finding a polygon of the decomposition**



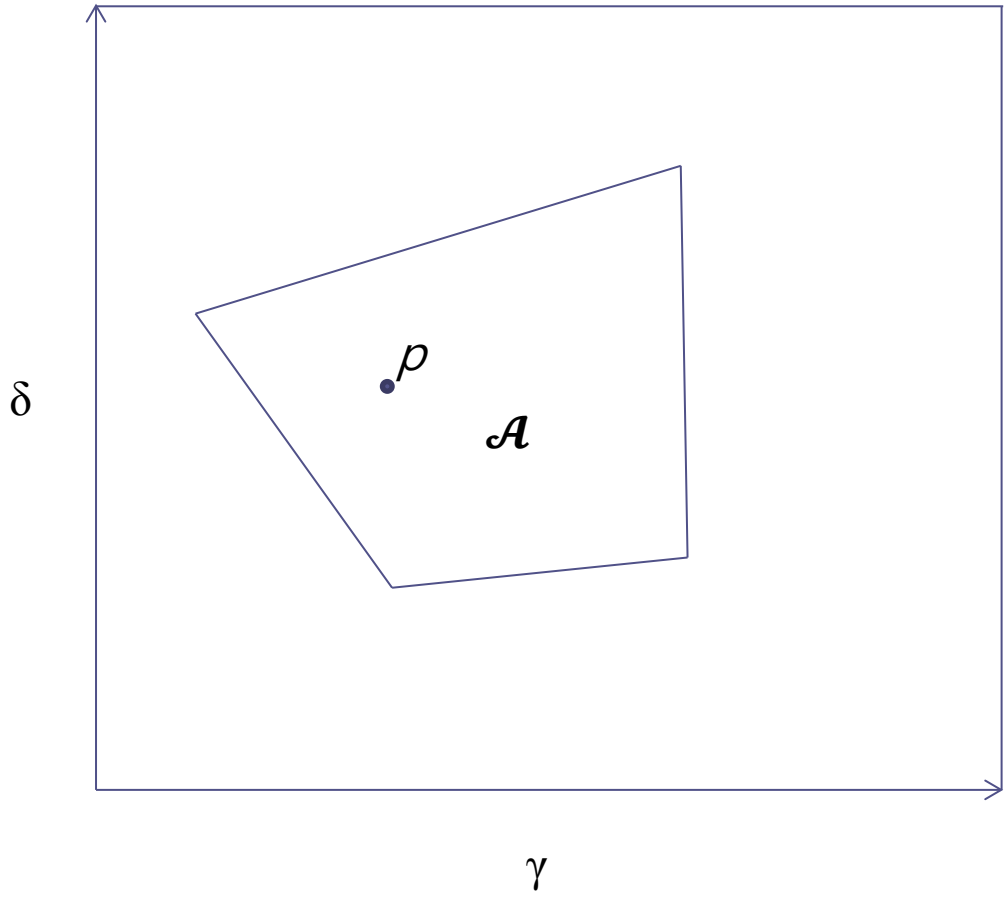
Efficient algorithms for computing a polygonal decomposition

- **Finding a polygon of the decomposition**



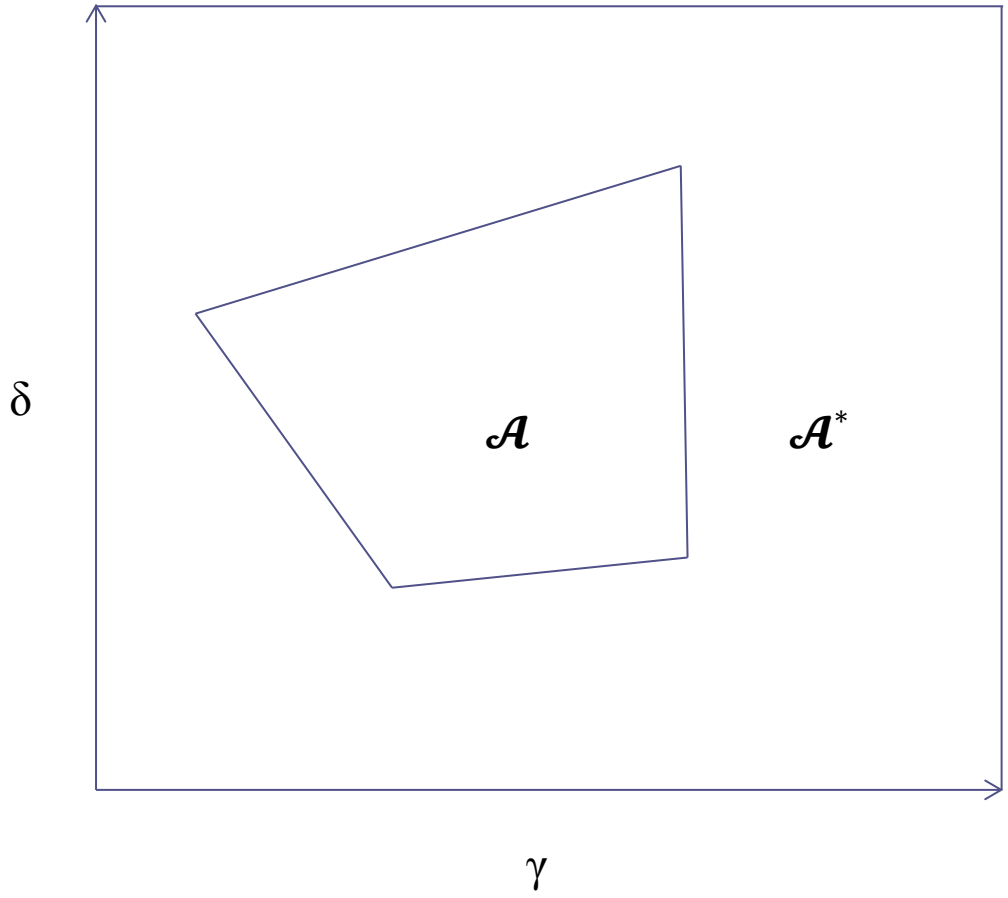
Efficient algorithms for computing a polygonal decomposition

- **Finding a polygon of the decomposition**



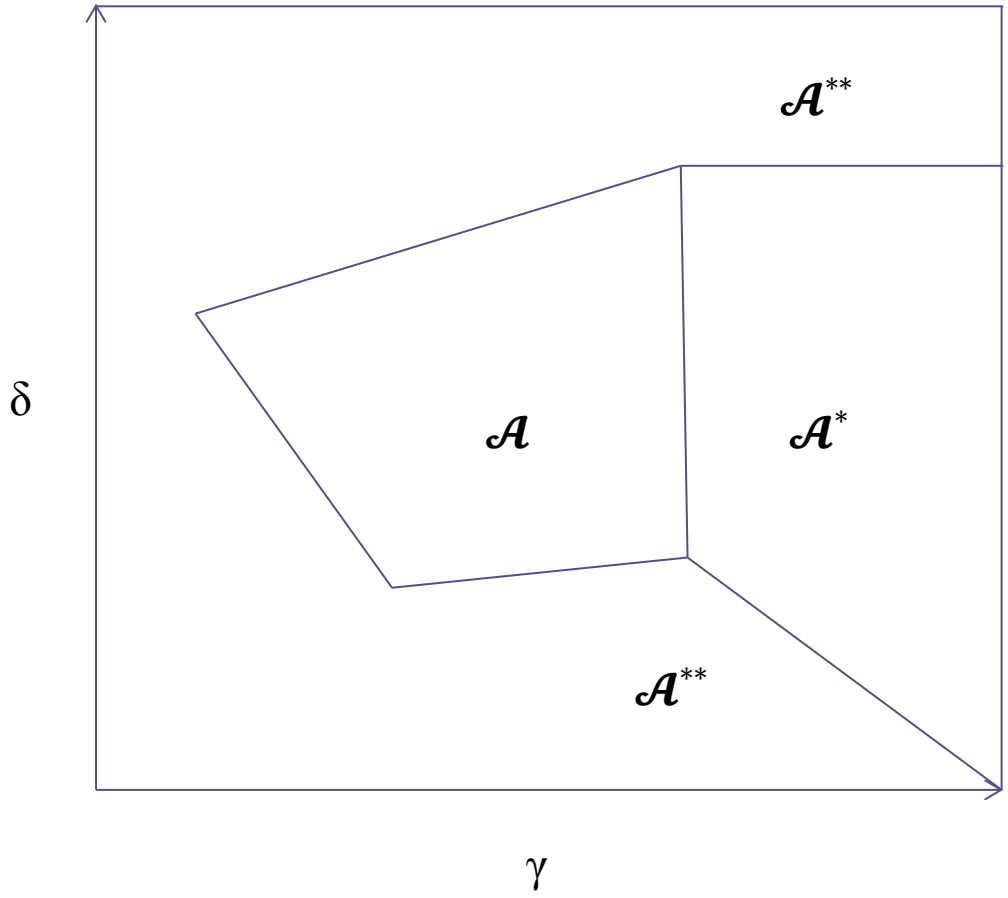
Efficient algorithms for computing a polygonal decomposition

- **Filling in the parameter space**



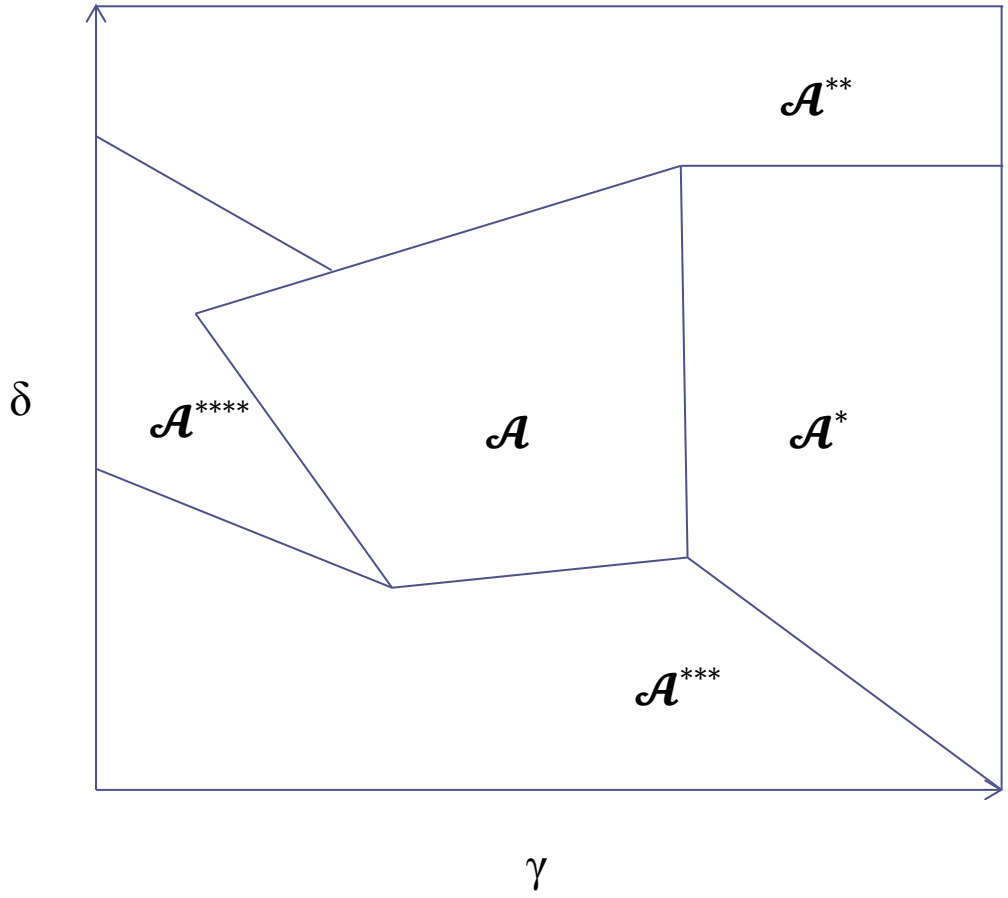
Efficient algorithms for computing a polygonal decomposition

- **Filling in the parameter space**



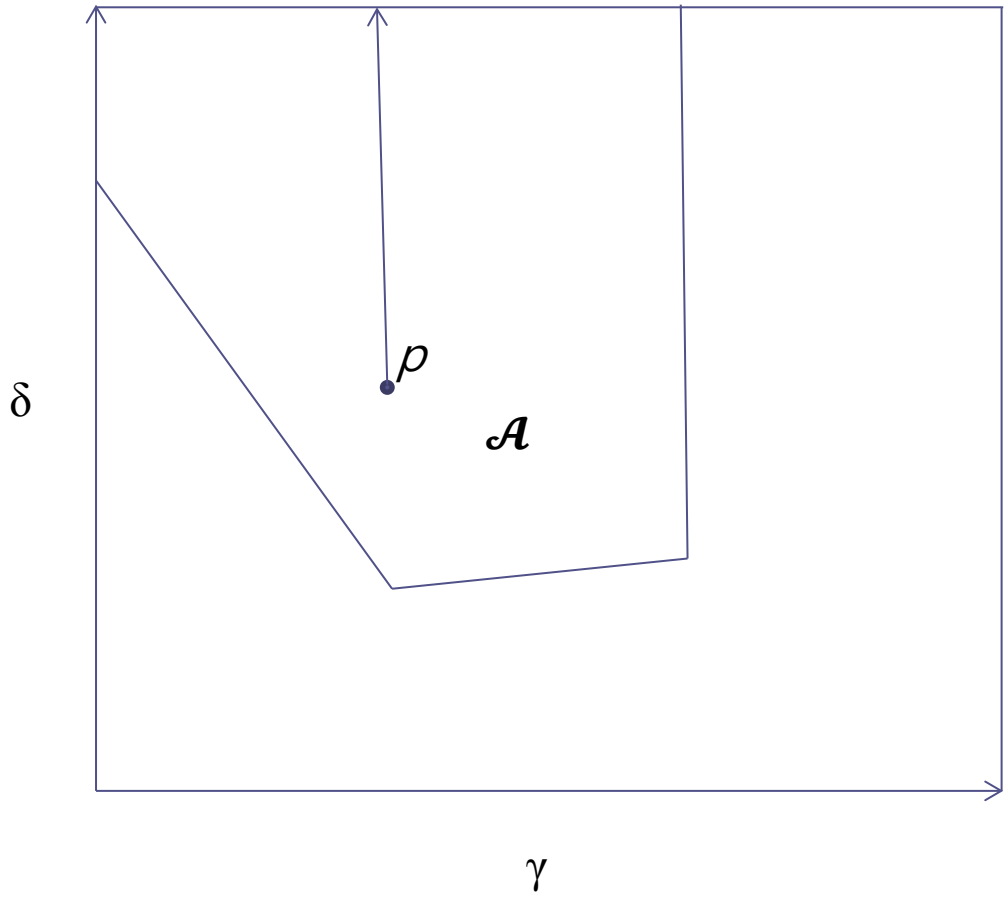
Efficient algorithms for computing a polygonal decomposition

- **Filling in the parameter space**



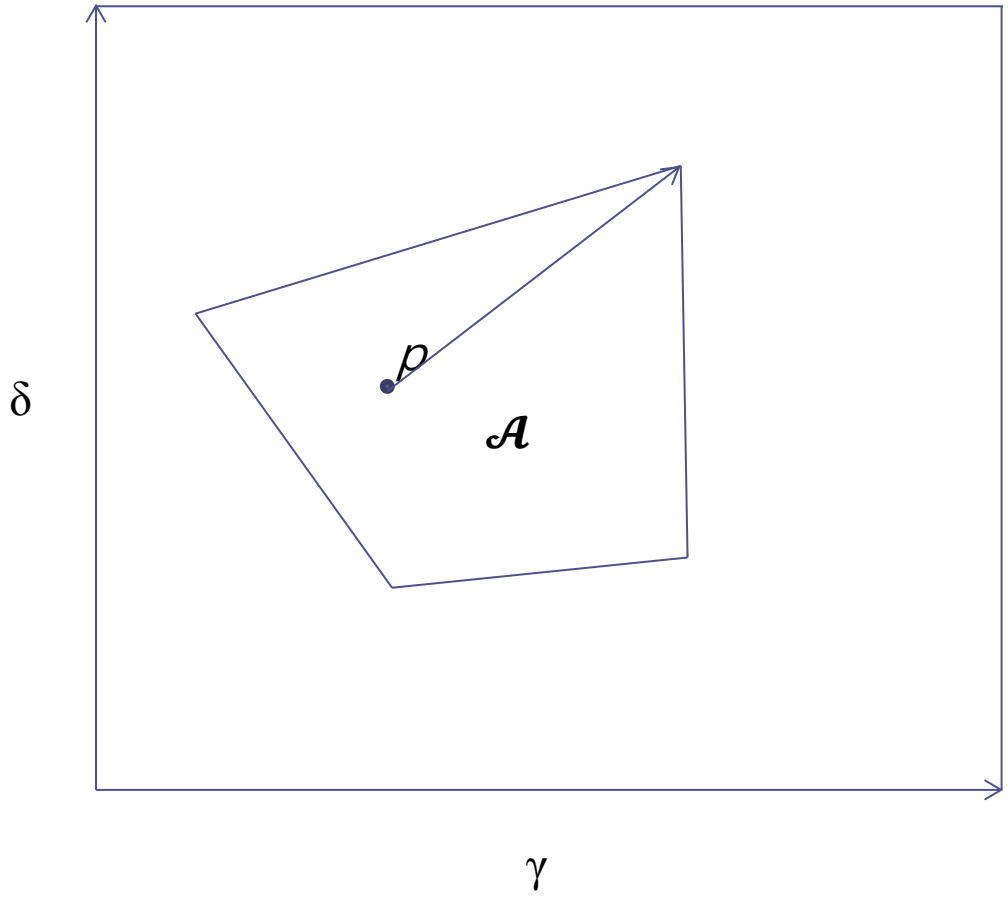
Efficient algorithms for computing a polygonal decomposition

- **Finding a polygon of the decomposition**



Efficient algorithms for computing a polygonal decomposition

- **Finding a polygon of the decomposition**



목차

- Introduction
- Definitions and first result
- Parametric alignment with the use of scoring matrices
- Efficient algorithms for computing a polygonal decomposition
- **Time analysis and the next idea**
- **Bounding the number of polygons in the decomposition**
- **Uses for parametric alignment**

Time analysis and the next idea

- **Let R, E, and V be the number of polygon, edges, and vertices. And let $O(nm)$ be the time to compute a single fixed-parameter alignment for sequence of lengths n and $m>n$.**

- **And let d be the number of edges of $\mathcal{P}(\mathcal{A})$. Then $3d$ ray searches are done to find the edges of $\mathcal{P}(\mathcal{A})$.**

Time analysis and the next idea

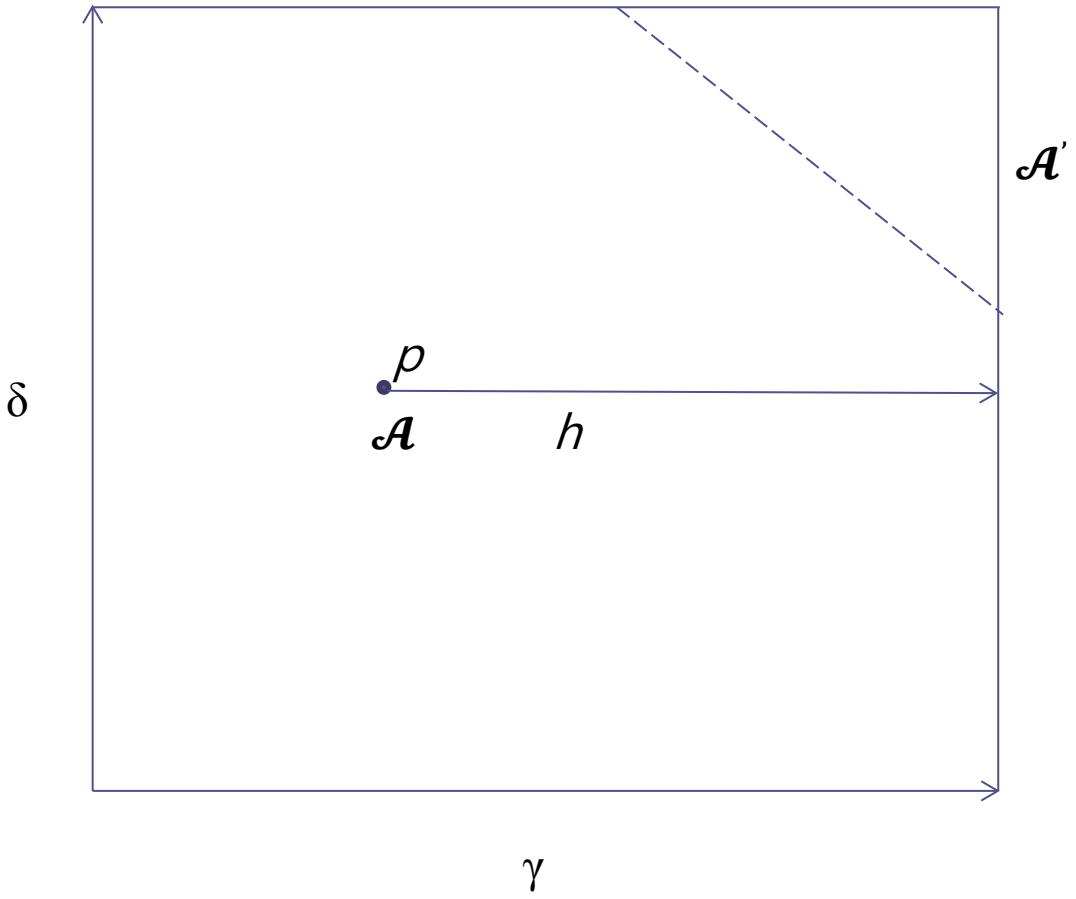
- **Because of degenerate case, at most 6*d* ray searches suffice to describe $\mathcal{P}(\mathcal{A})$. Each edge lies on at most two polygons, so the algorithm does at most 12*E* ray searches to find the complete decomposition.**

- **From Newton's third law each ray search requires at most *R* fixed-parameter alignment computations, so the complete decomposition requires at most 12*RE* fixed-parameter alignments, which can be don in *O*(*ERnm*) time.**

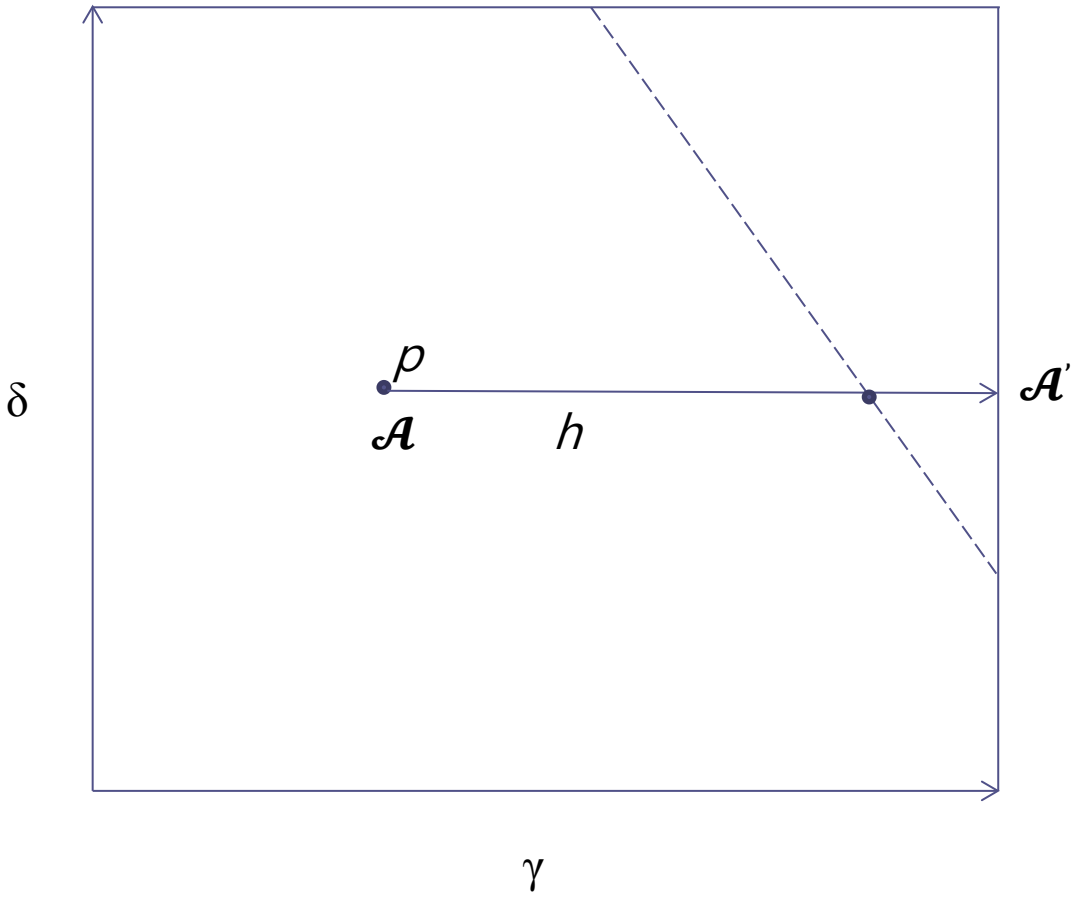
Time analysis and the next idea

- **Modified Newton's algorithm**
  - Consider any alignment $\mathcal{A}$' computed before the present execution of Newton's method.
  - Compute the intersection of the plane for $\mathcal{A}$ and $\mathcal{A}$' and project that line onto the $\gamma-\delta$ plane

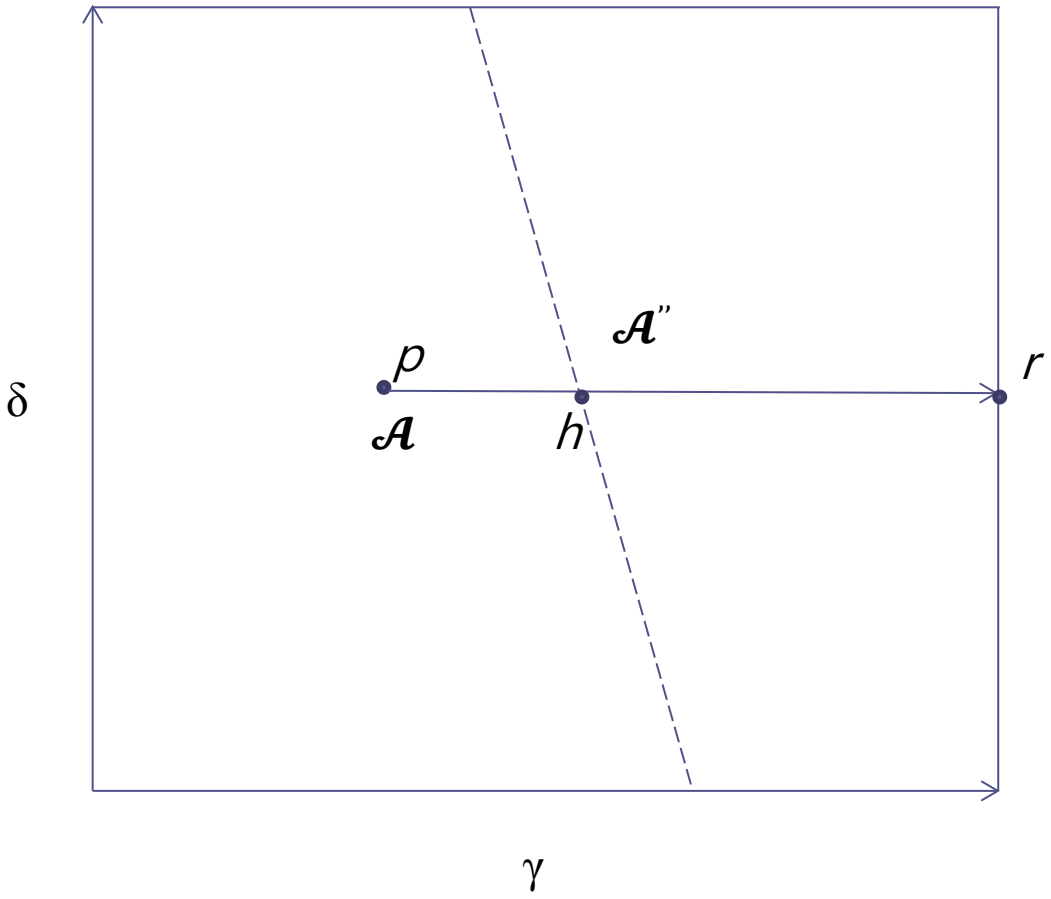
Time analysis and the next idea



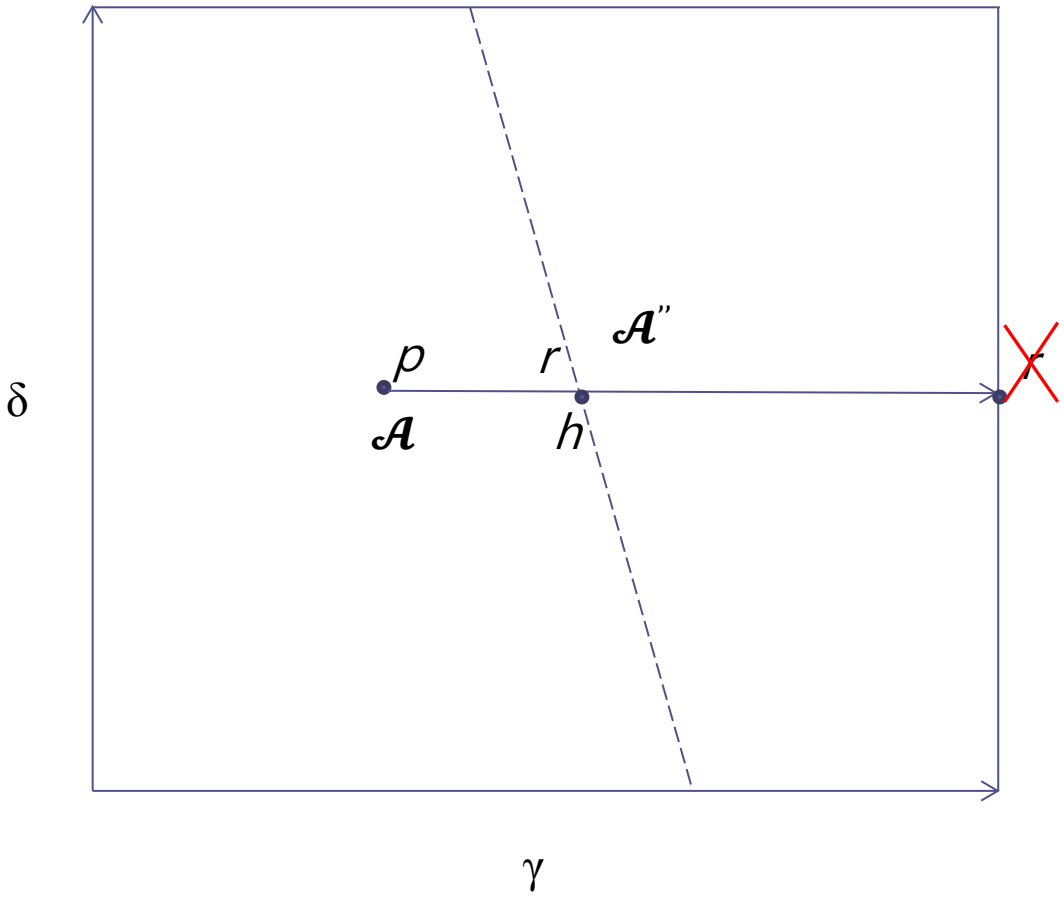
Time analysis and the next idea



Time analysis and the next idea



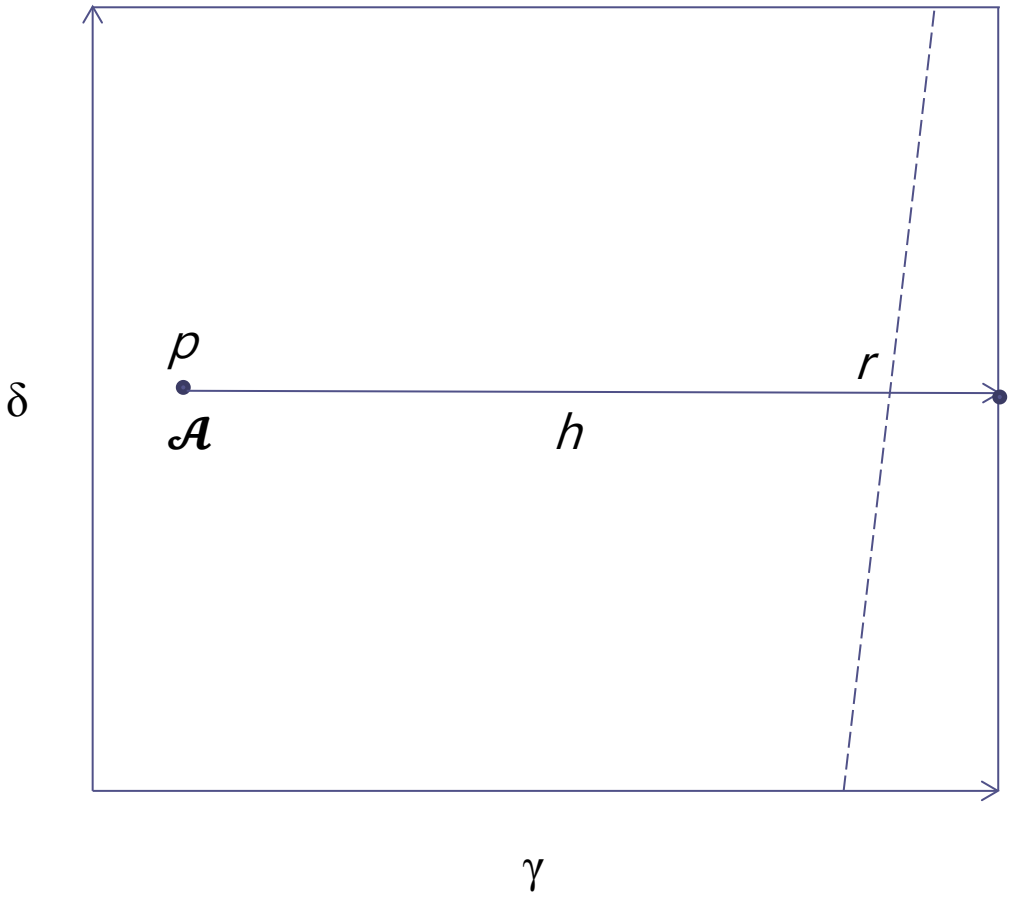
Time analysis and the next idea



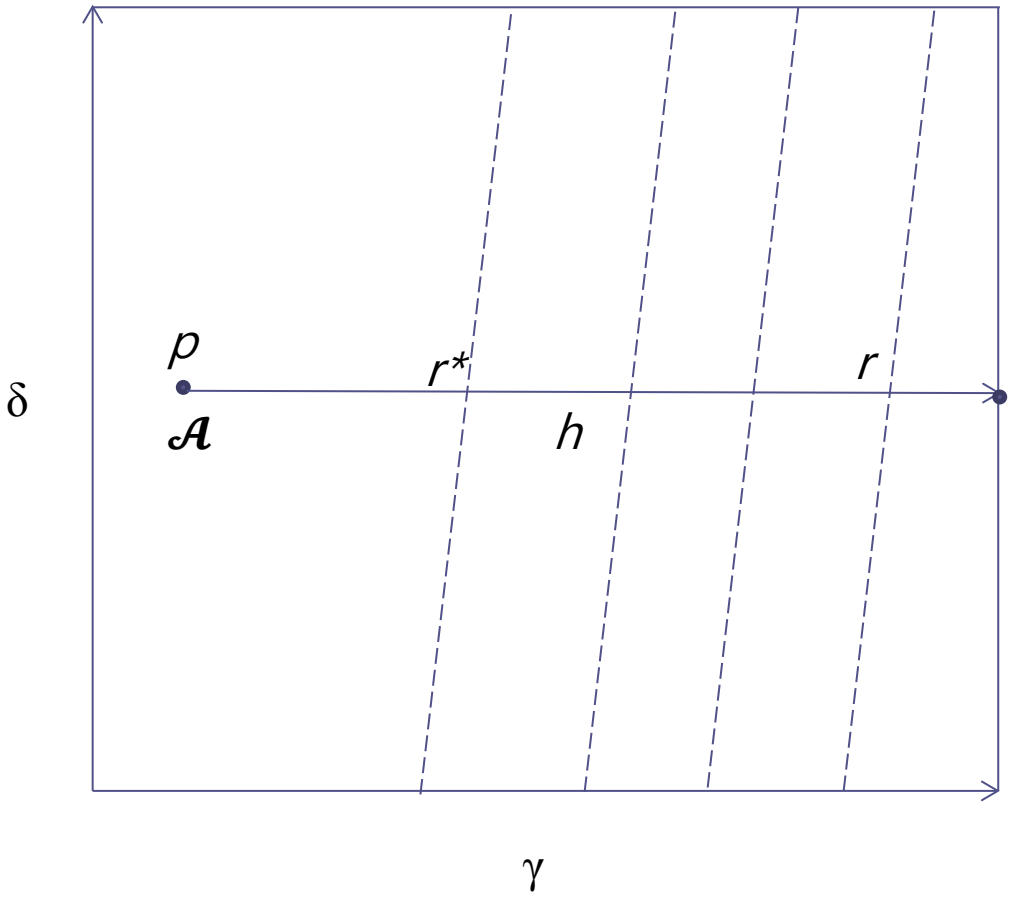
Time analysis and the next idea

- **The added bookkeeping time for using L' is just $O(V+E+R)$ per ray search or $O(12E(V+E+R))$ overall**
  - Whenever an alignment is computed, the vector for that alignment is placed into *L'* (if it isn't already there), whether or not that alignment is optimal for a polygon, for an edge or only for a vertex.

Time analysis and the next idea



Time analysis and the next idea



Time analysis and the next idea

- **Use of the modified Newton's method does not change the fact that at most 12*E* ray searches are done.**
  - One alignment computation per ray search is *redundant*.
  - So each of the other fixed-parameter alignment computation must find a new vector to add to *L'*, and size of L' is at most $V+E+R$
  - It follows that the complete polygonal decomposition is computed using at most $V+13E+R$ fixed-parameter alignments.

Time analysis and the next idea

- **Overall time bound of modified Newton's algorithms is $O(12E(V+E+R)+(V+13E+R)nm)$**
  - Each vertex is incident with at least three edges, $V \leq E \leq 3R$.
  - This is easy to show using Euler's theorem on planar graphs.

- **Thus the terms $12E$, $V+E+R$, and $V+13E+R$ are each $O(R)$ Hence, the above time bound becomes $O(R^2+Rnm)$, which is $O(R+nm)$ per polygon**

Time analysis and the next idea

- **If we use some form of alignment where the fixed-parametric computation takes $O(C)$ time, rather than $O(nm)$ time, then the time bound is $O(R+C)$ per polygon.**
  - However, we will show below that when no character-specific scoring matrices are used, then $R = O(nm)$.
  - When global alignment is computed and no scoring matrices are used, then $R<n^{2/3}$.
  - When scoring matrices are used, but γ and δ are the chosen variable parameters, then again $R = O(nm)$.

Time analysis and the next idea

- **Theorem 13.1.2**
  - For most of the (important) parameter choices, a full polygonal decomposition can be found in O(nm) time per polygon (i.e., proportional to the time needed to compute just a single fixed-parameter alignment)

목차

- Introduction
- Definitions and first result
- Parametric alignment with the use of scoring matrices
- Efficient algorithms for computing a polygonal decomposition
- Time analysis and the next idea
- **Bounding the number of polygons in the decomposition**
- **Uses for parametric alignment**

Bounding the number of polygons in the decomposition

- **How many polygons there can be in various types of polygonal decompositions?**
  - It addresses the concern that there may be so many polygons that the decomposition will give no interesting information.
  - Because the polygonal decomposition method discussed in the previous section runs in $O(R+nm)$ time per polygon, we must bound R to establish Theorem 13.1.2

- **Theorem 13.1.3**
  - Consider the following scoring scheme that does not use scoring matrices : $\alpha * mt - \beta * ms - \gamma * id - \delta * gp$ no matter which two of the four parameters are chosen to be variable, and no matter what type of alignment(global, local, etc..) is computed, the polygonal decomposition can contain at most $O(nm)$ polygons

Bounding the number of polygons in the decomposition

- **Proof**
  - For illustration, suppose γ and δ are variable and that α and β are fixed at α_0 and β_0. Then for any alignment $\boldsymbol{\mathcal{A}}$, $\alpha_0 * mt - \beta_0 * ms$ is a constant, say $C_{\boldsymbol{\mathcal{A}}}$, and the value of $\boldsymbol{\mathcal{A}}$ is $C_{\boldsymbol{\mathcal{A}}} - \gamma * id_{\boldsymbol{\mathcal{A}}} - \delta * gp_{\boldsymbol{\mathcal{A}}}$. We associate $\boldsymbol{\mathcal{A}}$ with the triple$(C_{\boldsymbol{\mathcal{A}}}, id_{\boldsymbol{\mathcal{A}}}, gp_{\boldsymbol{\mathcal{A}}})$. Now if $\boldsymbol{\mathcal{A}}'$ is another alignment that has $id_{\boldsymbol{\mathcal{A}}}$ spaces and $gp_{\boldsymbol{\mathcal{A}}}$ gaps, and $C_{\boldsymbol{\mathcal{A}}} < C_{\boldsymbol{\mathcal{A}}'}$, then $\boldsymbol{\mathcal{A}}$ can never be optimal at any γ, δ point. Therefore, among all triples whose last two terms are $id_{\boldsymbol{\mathcal{A}}}$, $gp_{\boldsymbol{\mathcal{A}}}$, at most one of those triples is associated with an alignment that is optimal at some point. If $n<m$ are the lengths of the two strings, then there can be at most $n+1$ gaps and $m+n$ spaces in an alignment of the two strings, so there are at most $O(nm)$ triples associated with an optimal alignment. The theorem follows because any two alignments associated with the same triple are optimal at exactly the same point.

Bounding the number of polygons in the decomposition

- **Lemma 13.1.3**
  - For any alignment with corresponding vector (*mt*, *ms*, *id*) : *2mt + 2ms + id = N*, where *N = n+m* is the sum of the two sequence lengths. hence *mt+ms+id/2 = N/2* for any global alignment.

- **PROOF**
  - A match and mismatch involves two characters. Thus the total number of characters that form part of a match or mismatch is 2(*mt+ms*)
  - An indel involves only one character from one sequence.
  - Each of the N characters is counted once as part of a match, a mismatch, or an indel.

Bounding the number of polygons in the decomposition

- **Corollary 13.1.2**
  - Every global alignment has the same value (N) at the point $\beta = -1, \gamma = -1/2$

- **PROOF**
  - Plugging into the objective function, we see that at point (-1,-1/2) every global alignment *CA* has value $mt_{\mathcal{A}} - ms_{\mathcal{A}} - id_{\mathcal{A}}/2$, which equals *N* by Lemma 13.1.3

Bounding the number of polygons in the decomposition

- **Theorem 13.1.4**
  - In the case of global alignment with no scoring matrices, there can be at most $O(n)$ polygons in the parametric decomposition, where $n \leq m$

- **PROOF**
  - Since each polygon boundary radiates from the point (-1,-1/2), each polygon boundary in the positive β, γ quadrant (the area of interest) must intersect either the horizontal or the vertical axis of the space. We will show that the number of intersections of the vertical axis cannot exceed n.

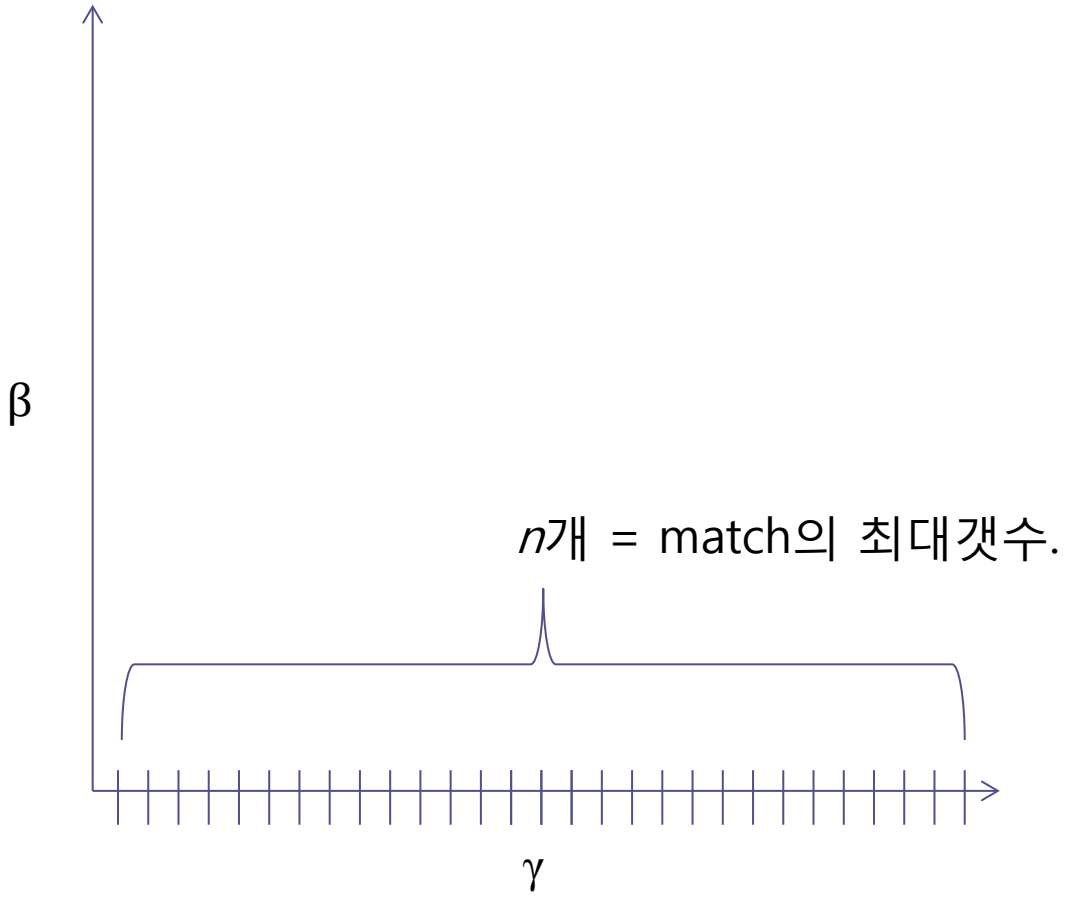
Bounding the number of polygons in the decomposition

- **PROOF**
  - Since each polygon boundary radiates from the point (-1,-1/2), each polygon boundary in the positive β, γ quadrant (the area of interest) must intersect either the horizontal or the vertical axis of the space. We will show that the number of intersections of the vertical axis cannot exceed n.

    Along the vertical (γ) axis, β is zero, so the parametric problem along that axis is to maximize $mt_{\mathcal{A}} - \gamma * id_{\mathcal{A}}$ as a function of the single parameter γ. Clearly, as γ increases, *mt* must decrease whenever the optimal changes (i.e., at each breakpoint along the γ axis). But since the number of matches can only vary from zero to n, there can be at most *n* polygon boundaries that intersect the γ axis.

    The same upper bound of n boundaries holds (by the same reasoning) along the horizontal axis. This proves the theorem.

Bounding the number of polygons in the decomposition



목차

- Introduction
- Definitions and first result
- Parametric alignment with the use of scoring matrices
- Efficient algorithms for computing a polygonal decomposition
- Time analysis and the next idea
- Bounding the number of polygons in the decomposition
- **Uses for parametric alignment**

Uses for parametric alignment

- **Sensitivity analysis**
  - The sensitivity of an alignment at point *p* can be examined by determining how far *p* is from a polygon boundary of the parametric decomposition.
  - If *p* is contained in a polygon $\mathcal{P}$, then additional information can be obtaining from the size of $\mathcal{P}$ and from how much the alignment value changes over $\mathcal{P}$ and over the neighboring polygons of $\mathcal{P}$.

Uses for parametric alignment

- **Efficient computation of all cooptimals**
  - It is often important to compute all the cooptimal alignment, rather than just a single optimal alignment, and it shows how parametric alignment can make this a more tractable task.
  - Parametric alignment can find nonintegral weight.