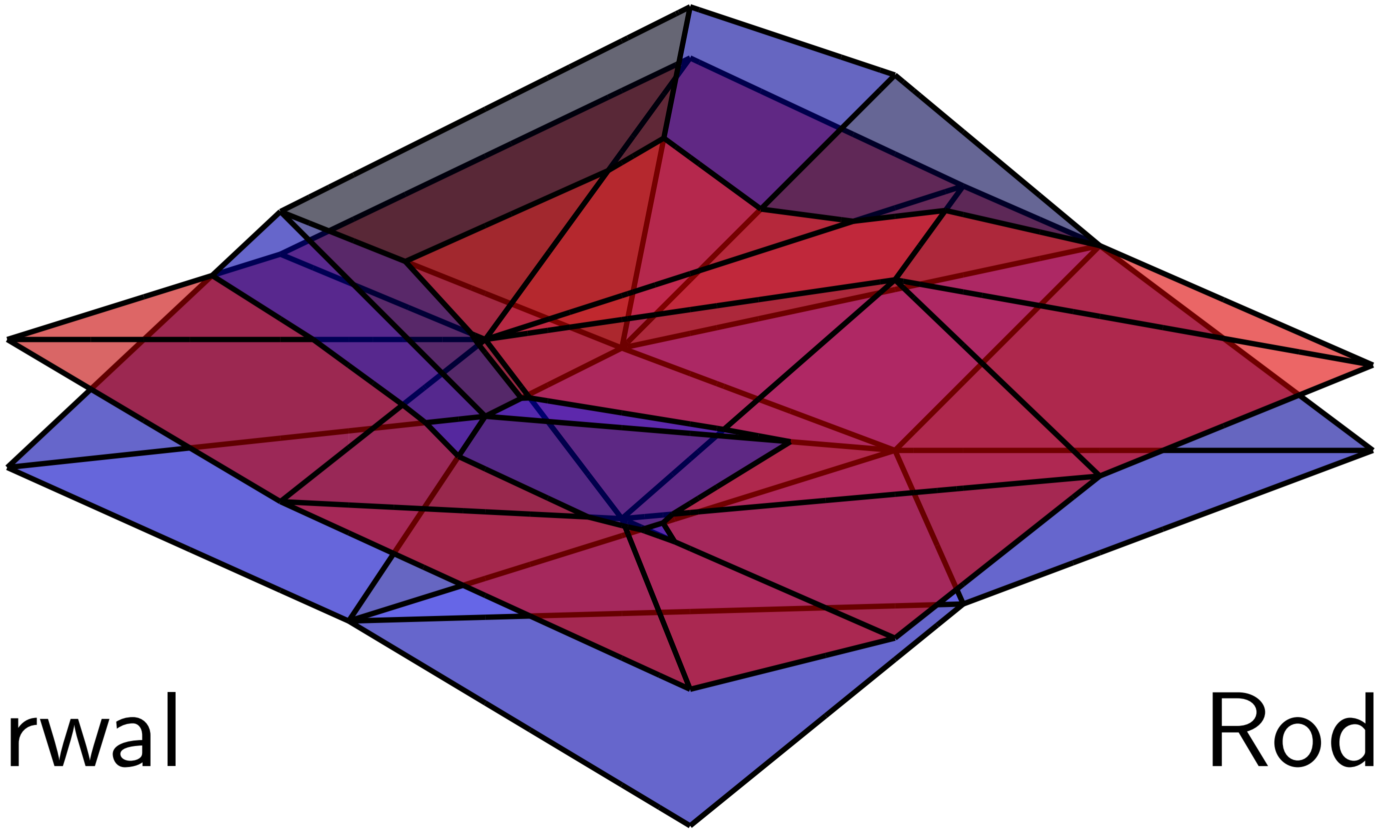
Matching Two Terrains under a Linear Transformation



Pankaj Agarwal

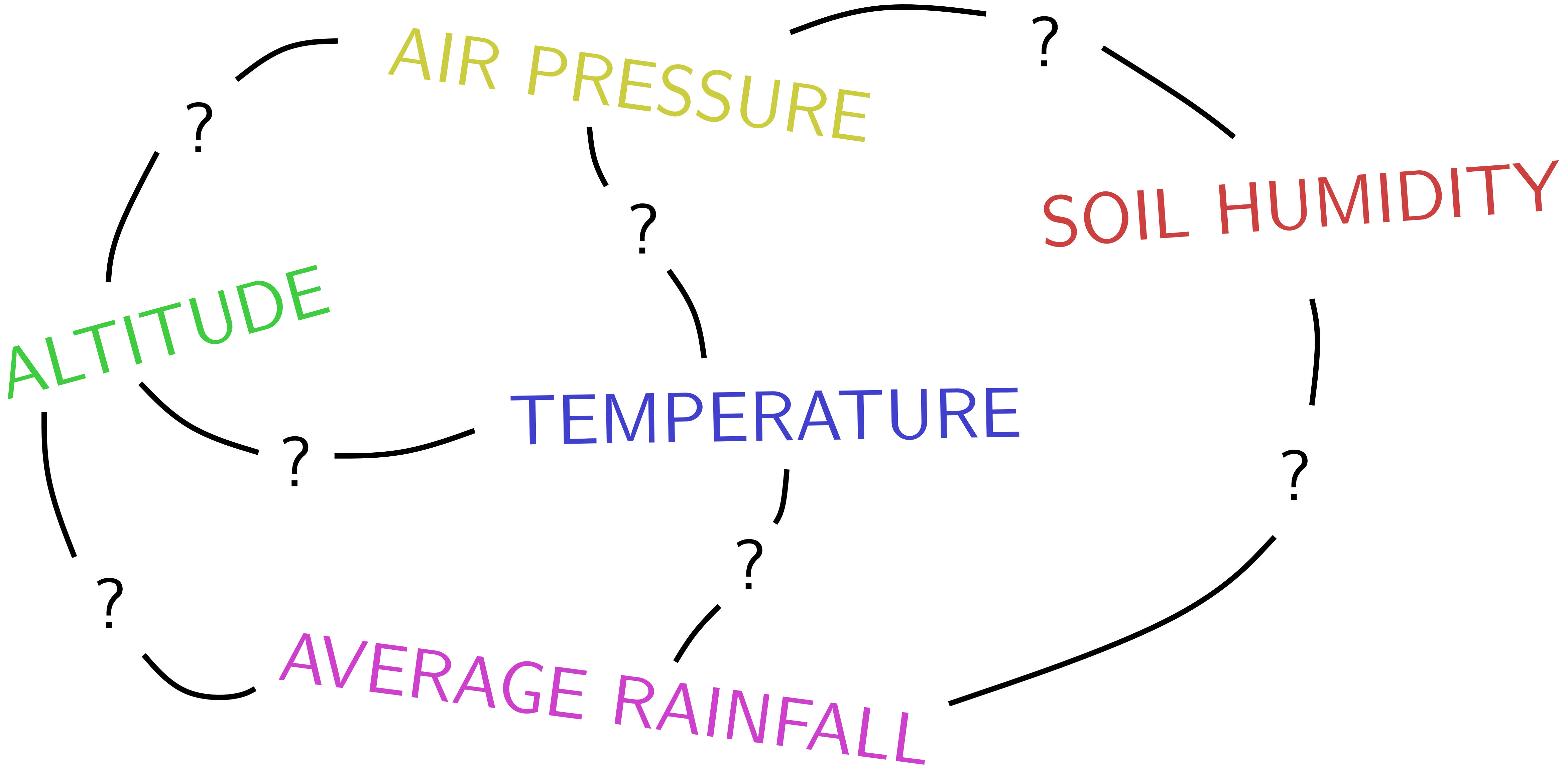
Rodrigo Silveira

Boris Aronov

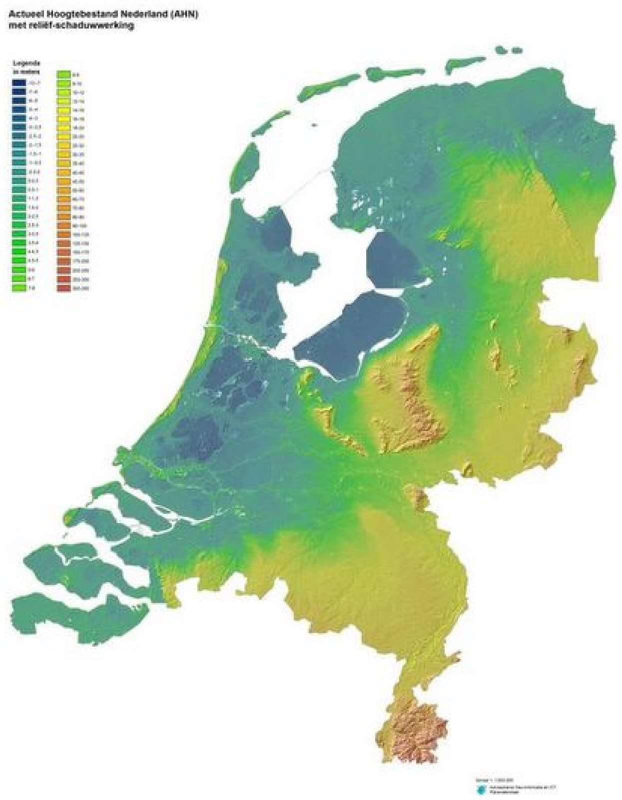
Maarten Löffler

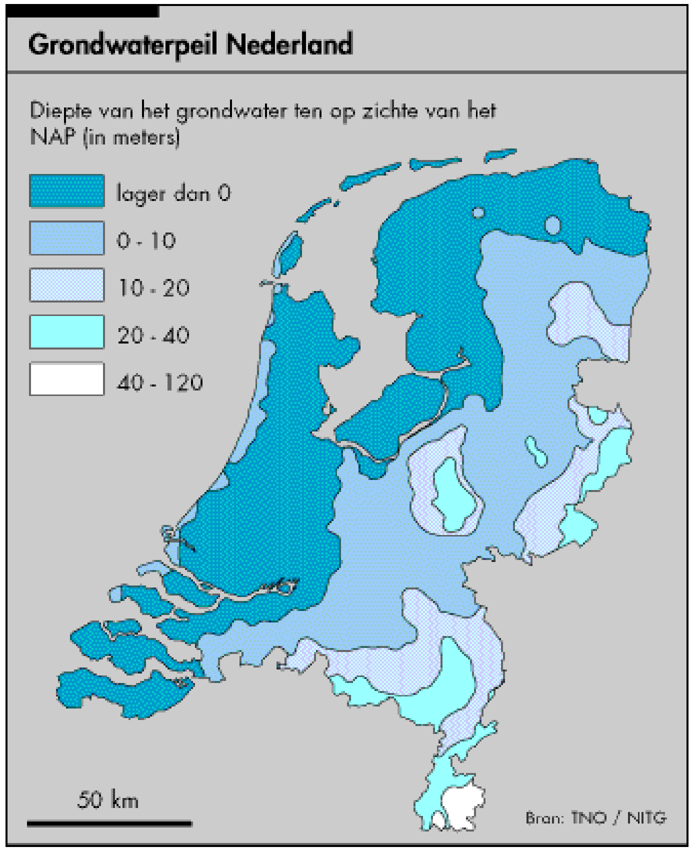
Marc van Kreveld

What is the problem?



We want to find linear dependencies between spatial data.

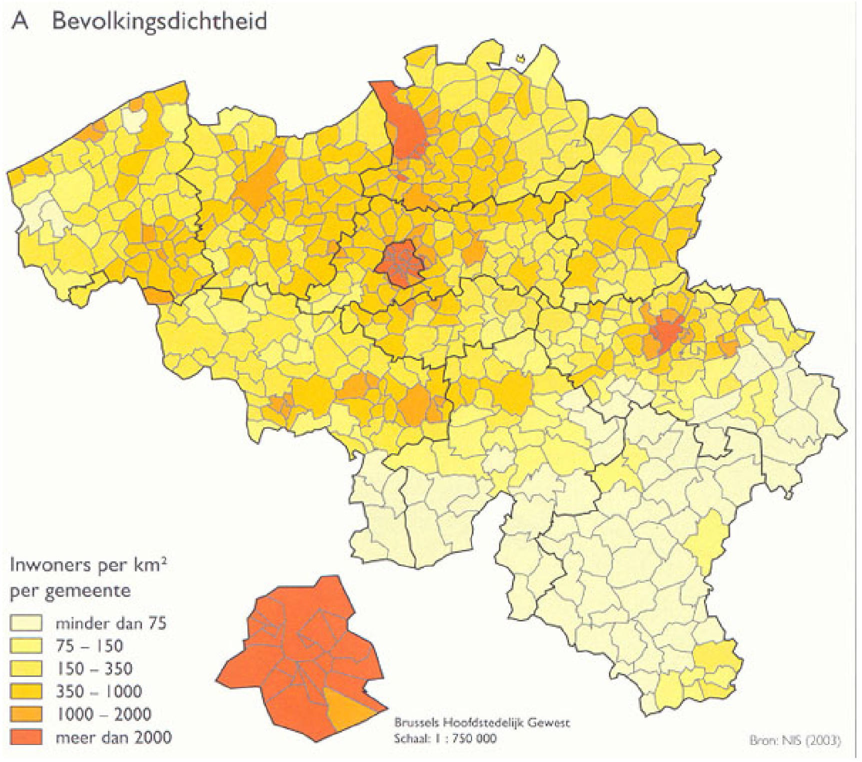




Source: Actueel Hoogtebestand Nederland

Source: TNO-NITG

For example, elevation above sea level in the Netherlands is probably related to groundwater level.



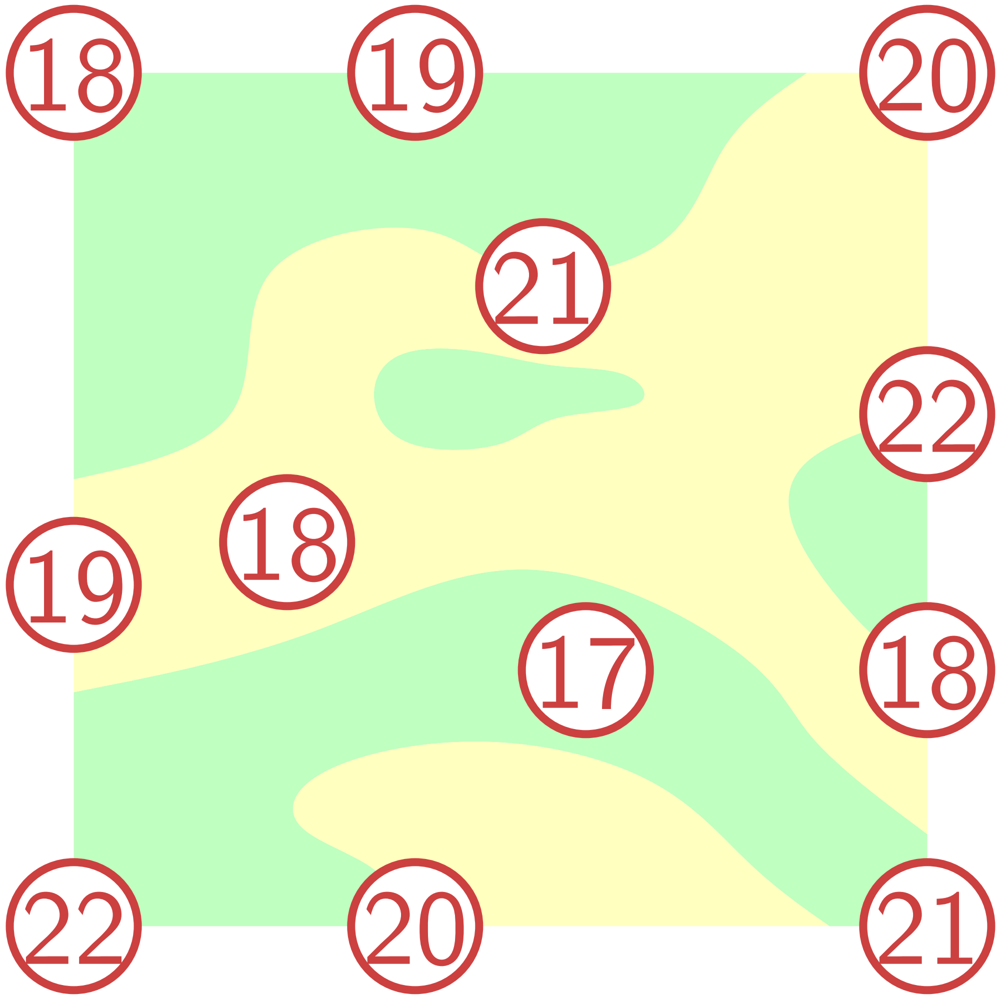
Source: Algemene Directie Statistiek België

Source: Wikipedia

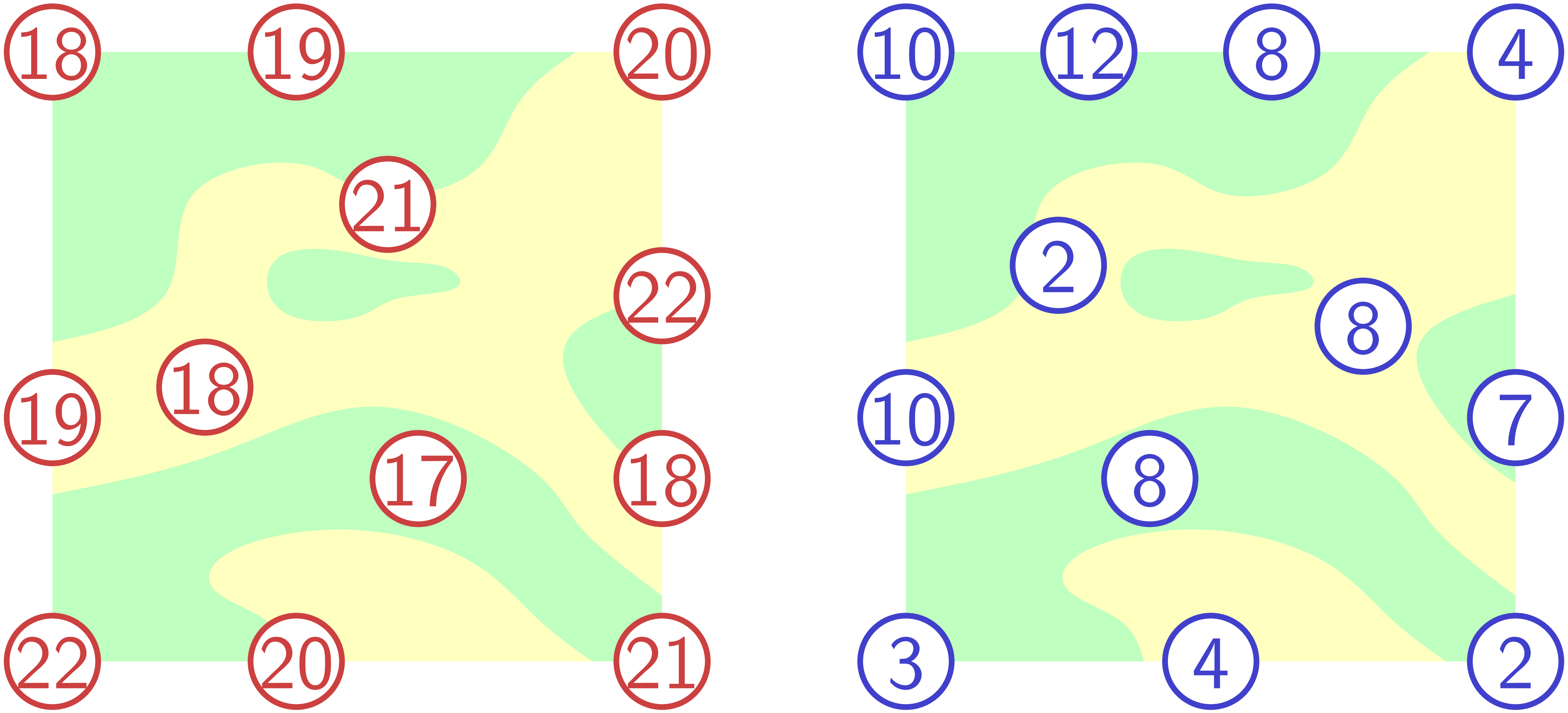
As another example, consider the population density and spoken language in Belgium.

Let's make this more formal.

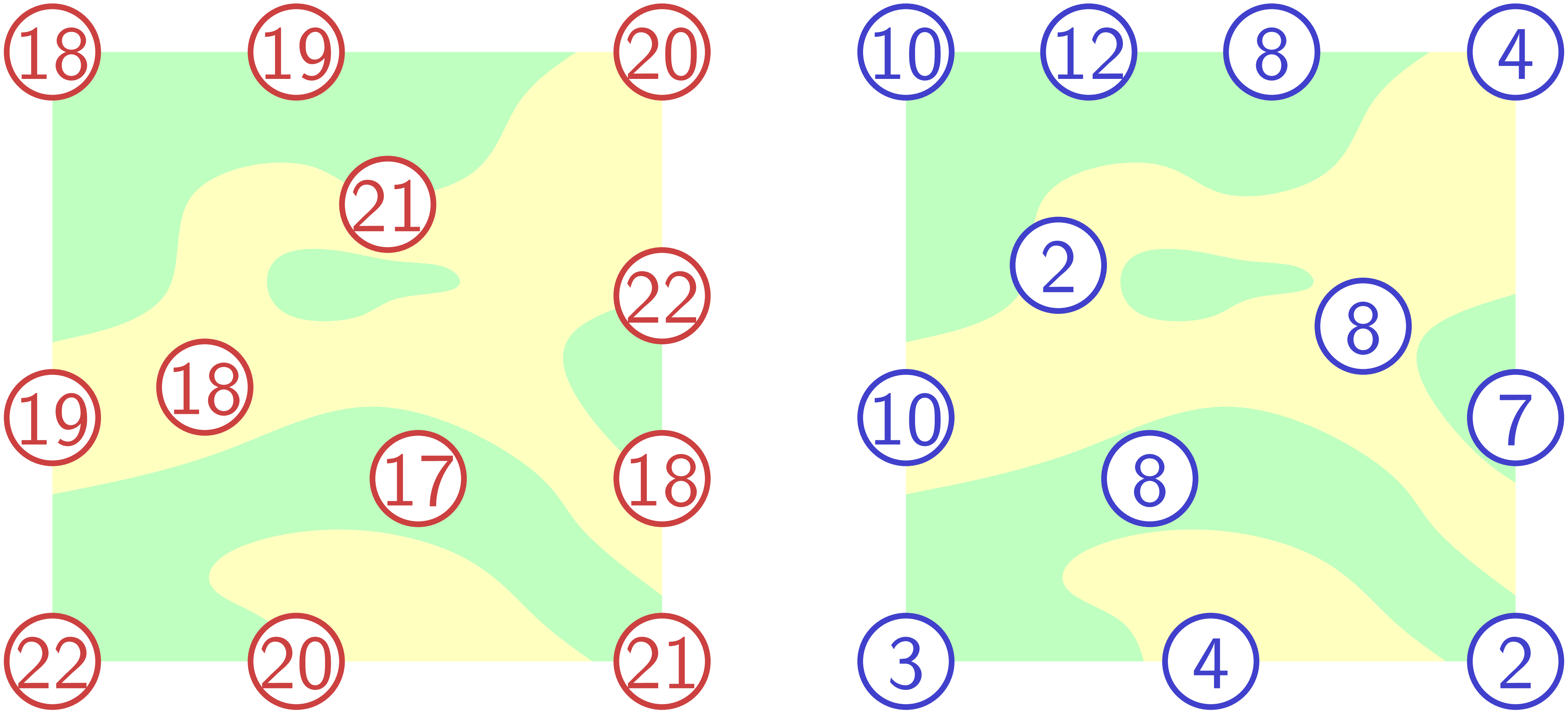
Consider some domain.



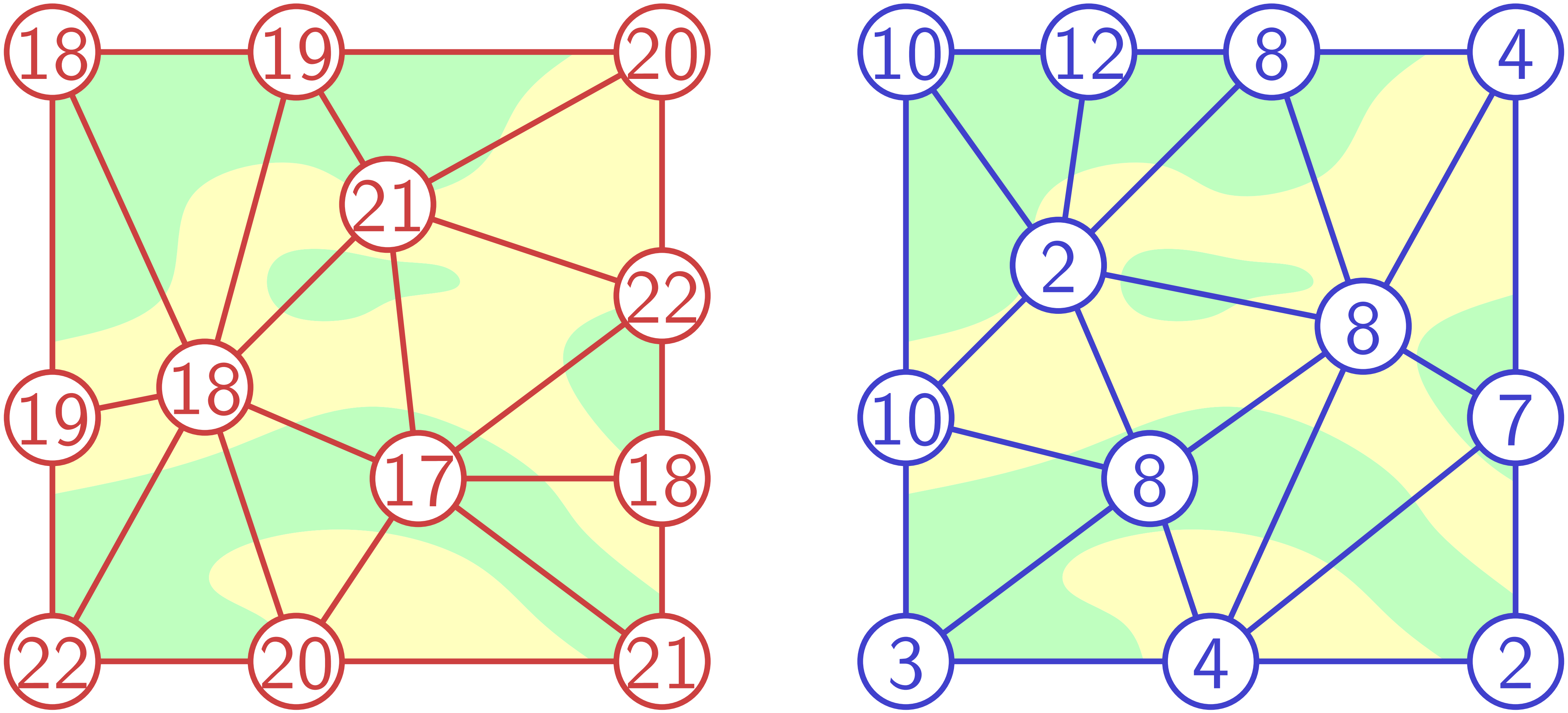
Suppose we measure some function f in several points on this domain.



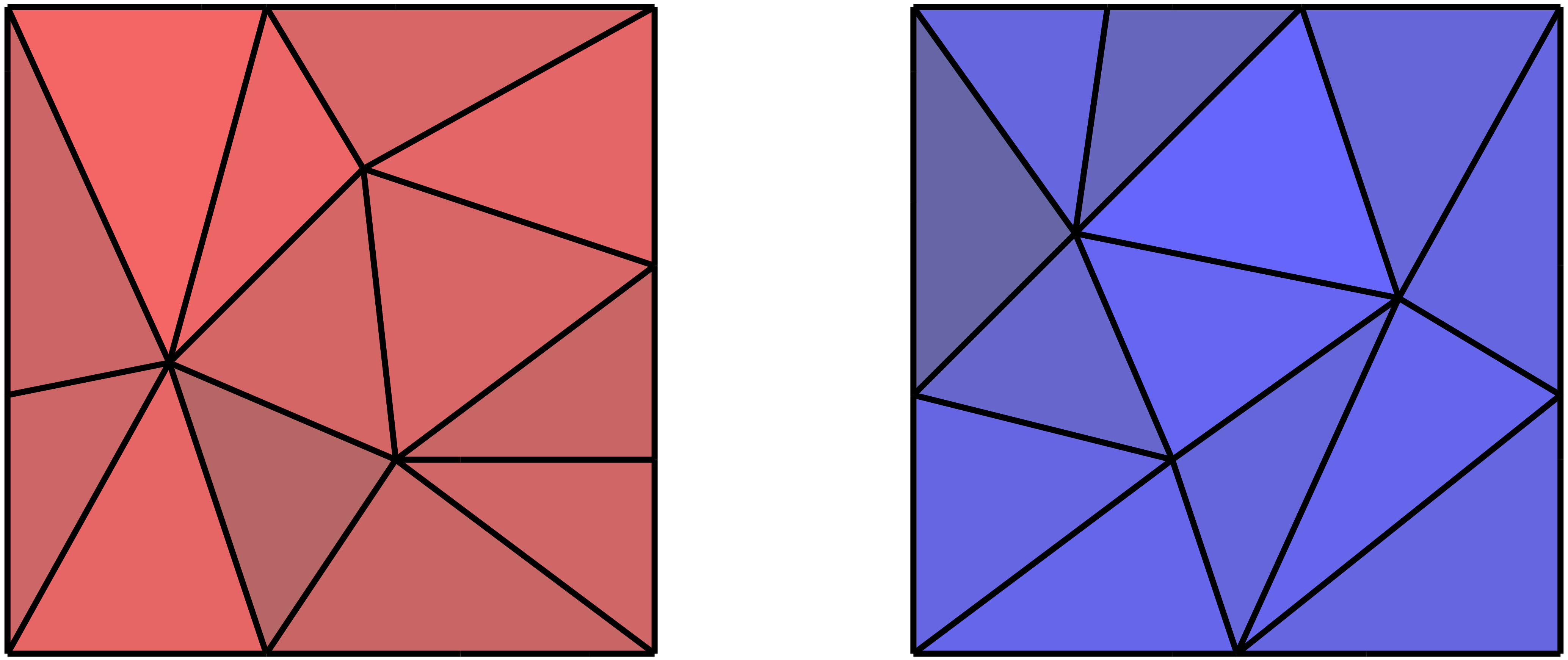
And suppose we also measure another function g on the same domain.



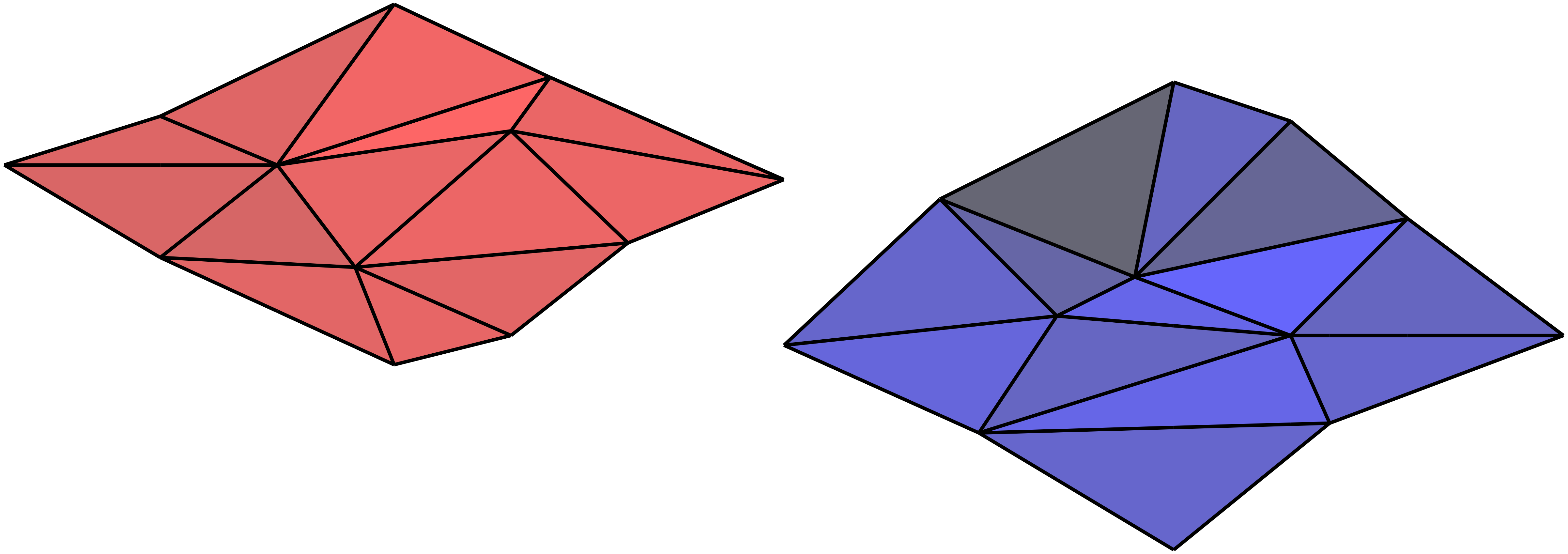
We want to know whether there exists a linear relationship between f and g.



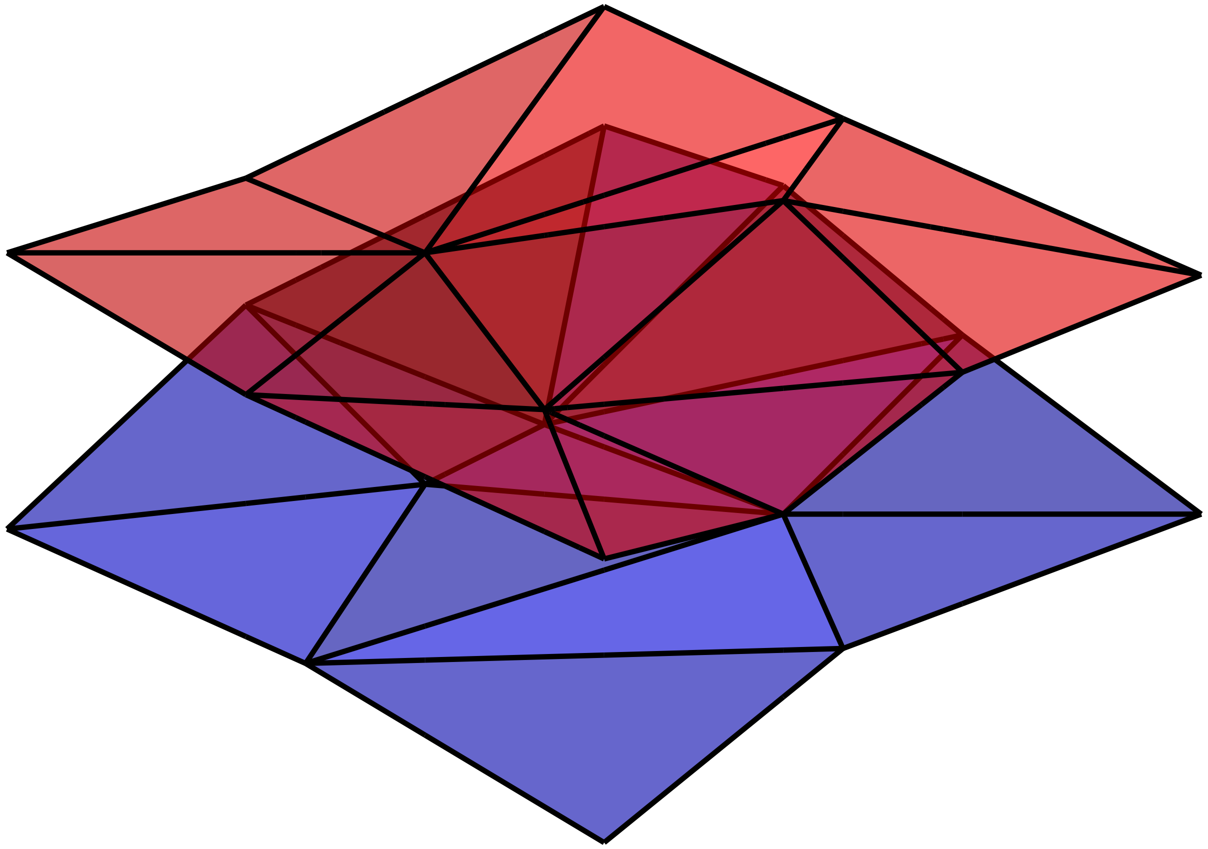
We estimate the functions on the whole domain by interpolation.



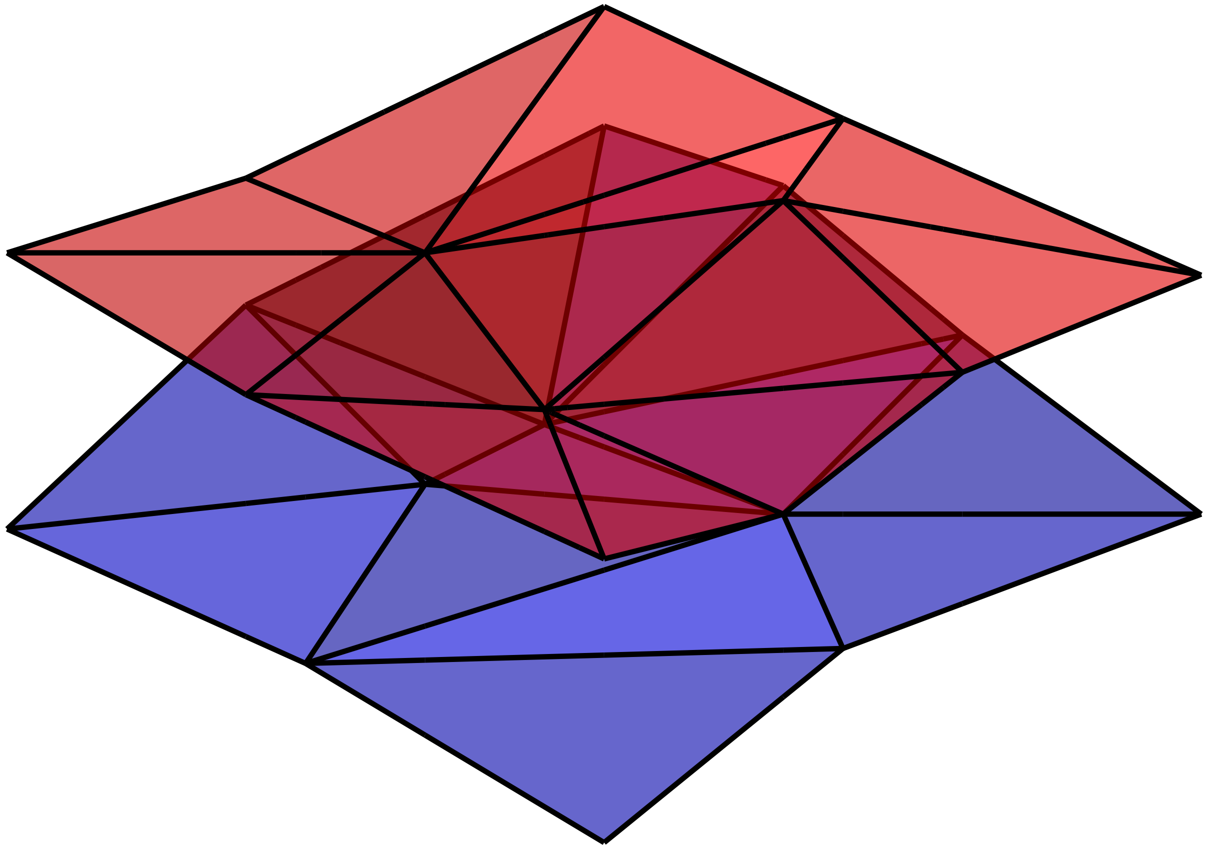
This results in two *Triangulated Irregular Terrains (TINs)* in 3D.



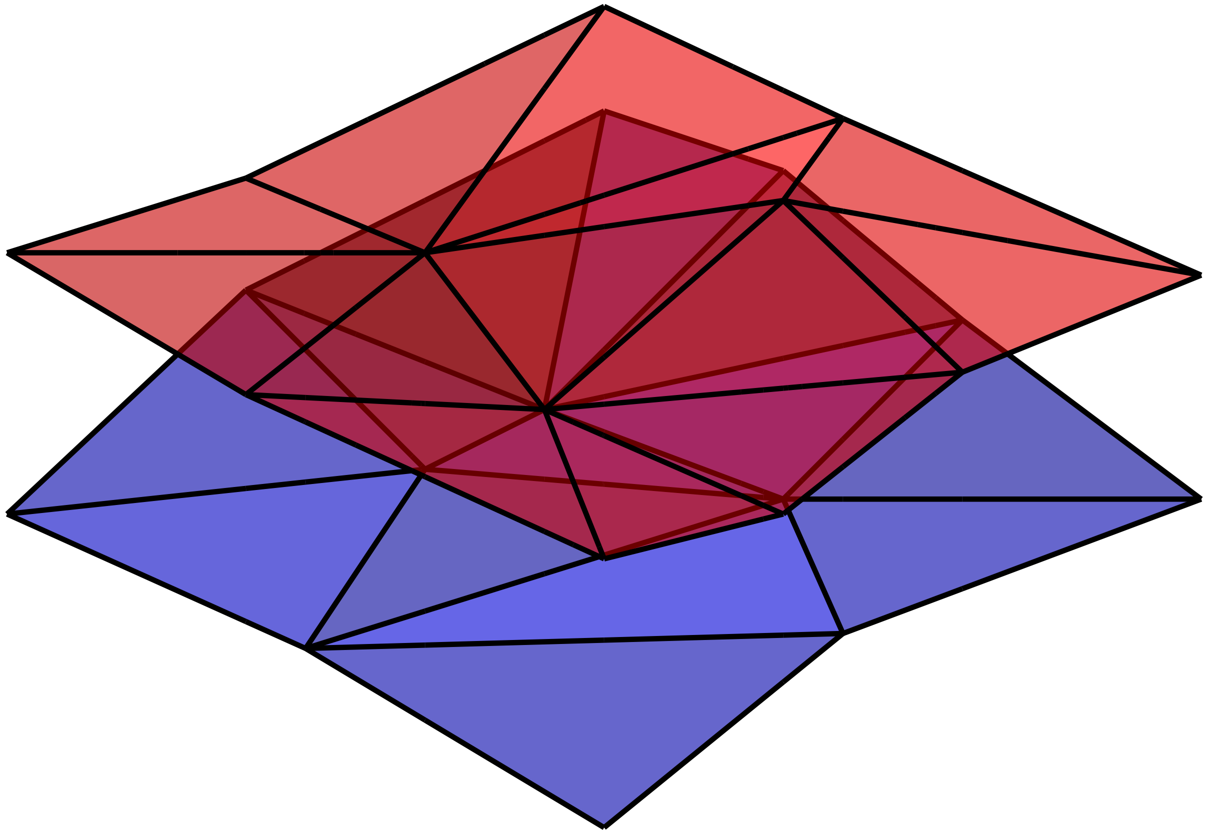
This results in two *Triangulated Irregular Terrains (TINs)* in 3D.



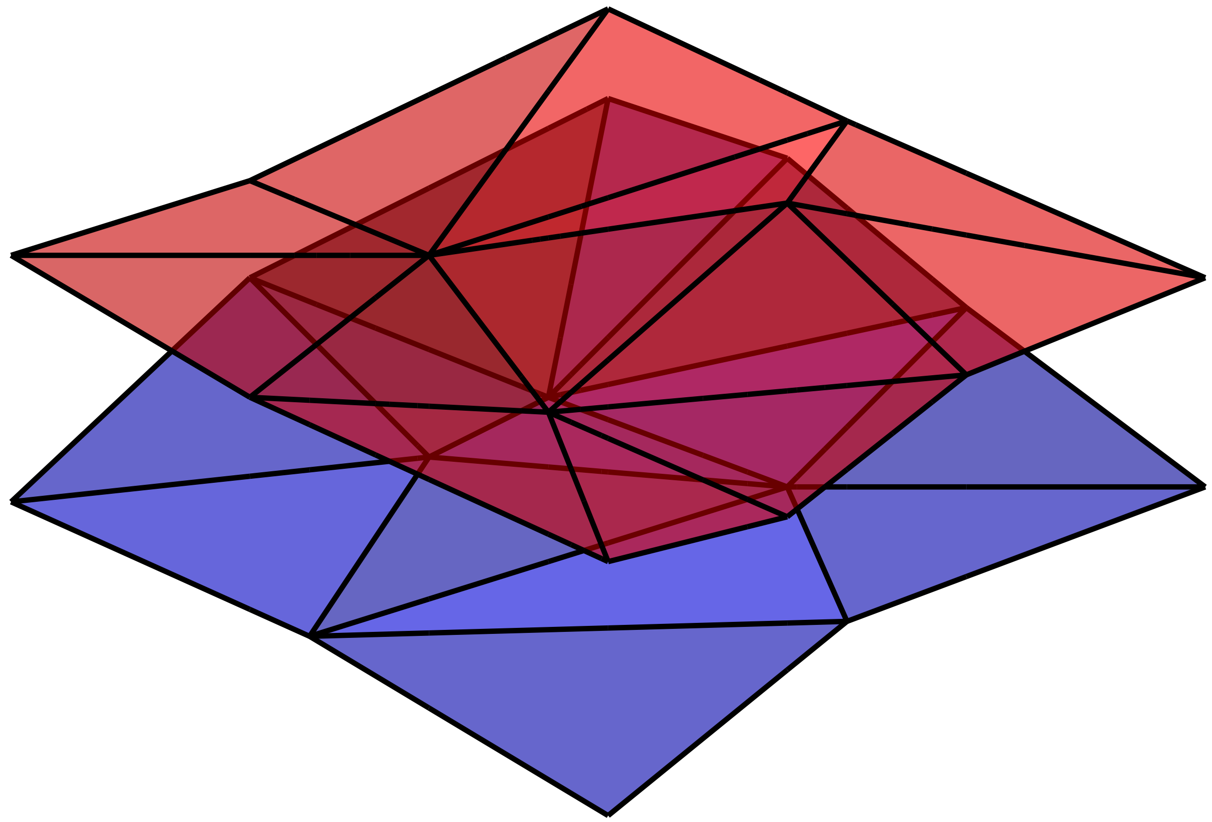
This results in two *Triangulated Irregular Terrains (TINs)* in 3D.



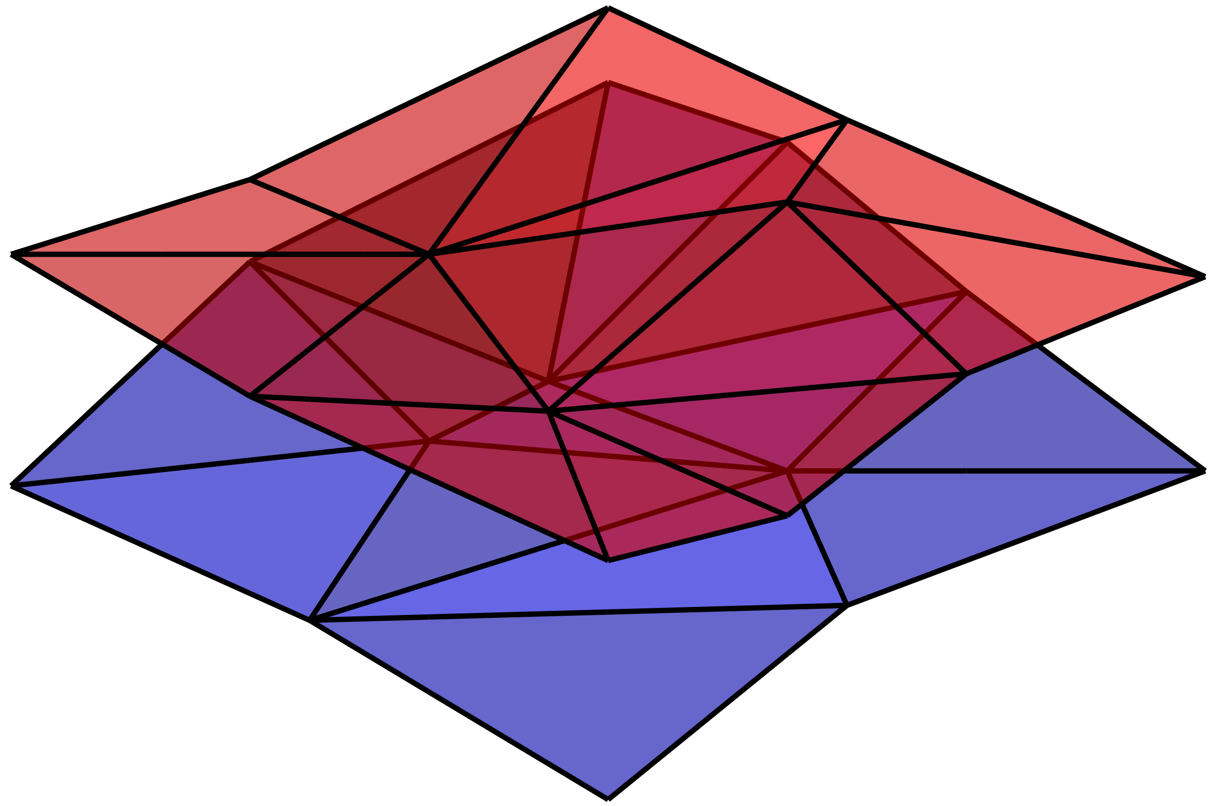
We want to find the translation and scaling of f to align it with g as good as possible.



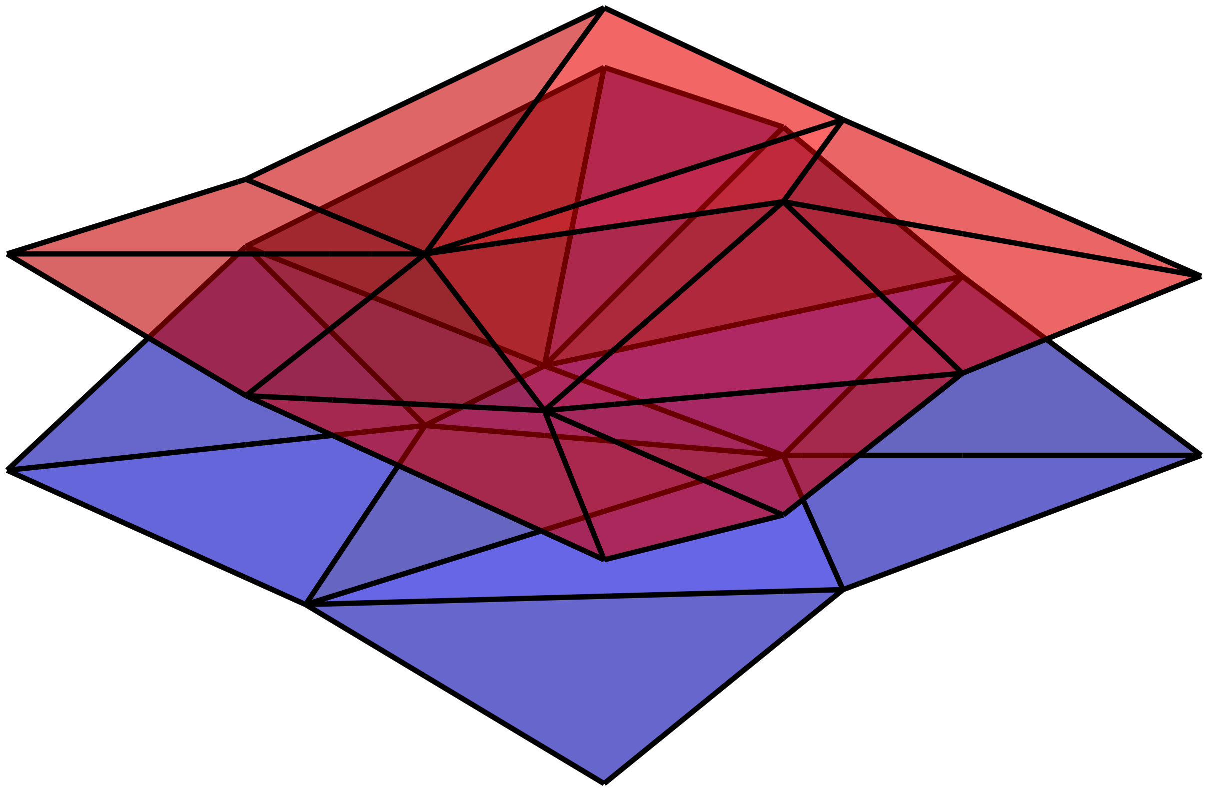
We want to find the translation and scaling of f to align it with g as good as possible.



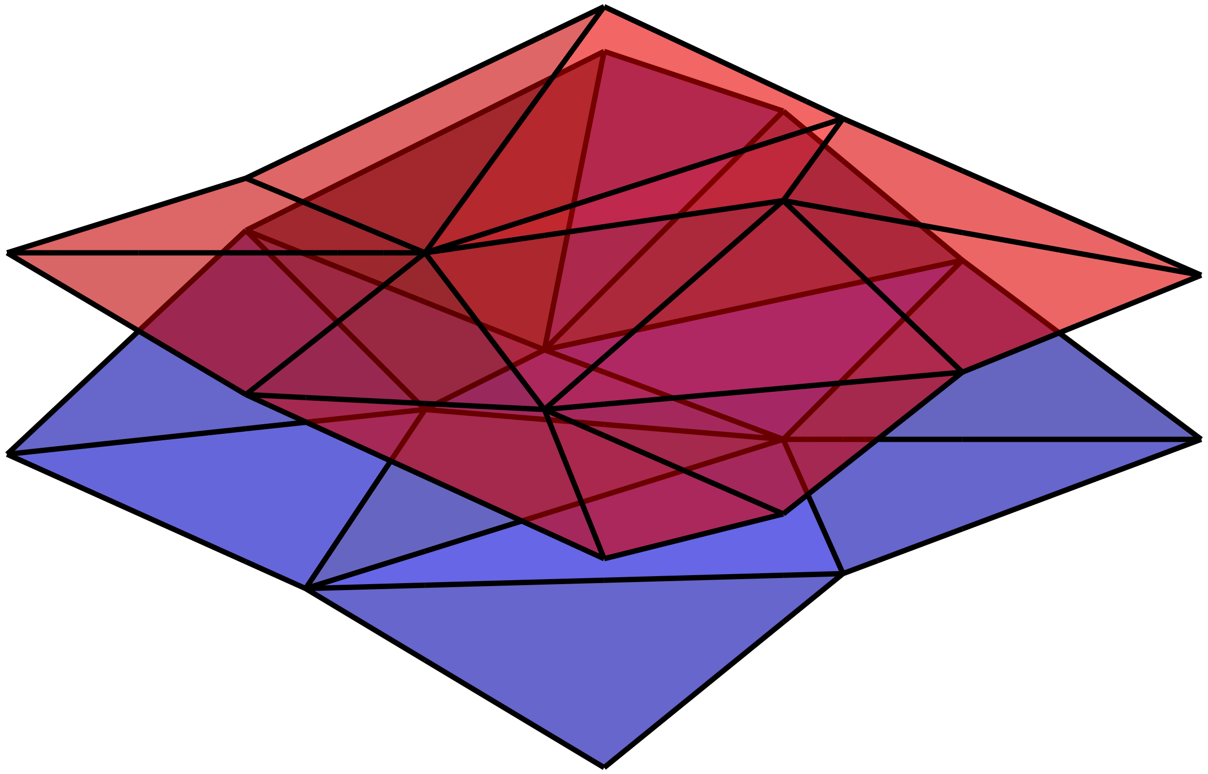
We want to find the translation and scaling of f to align it with g as good as possible.



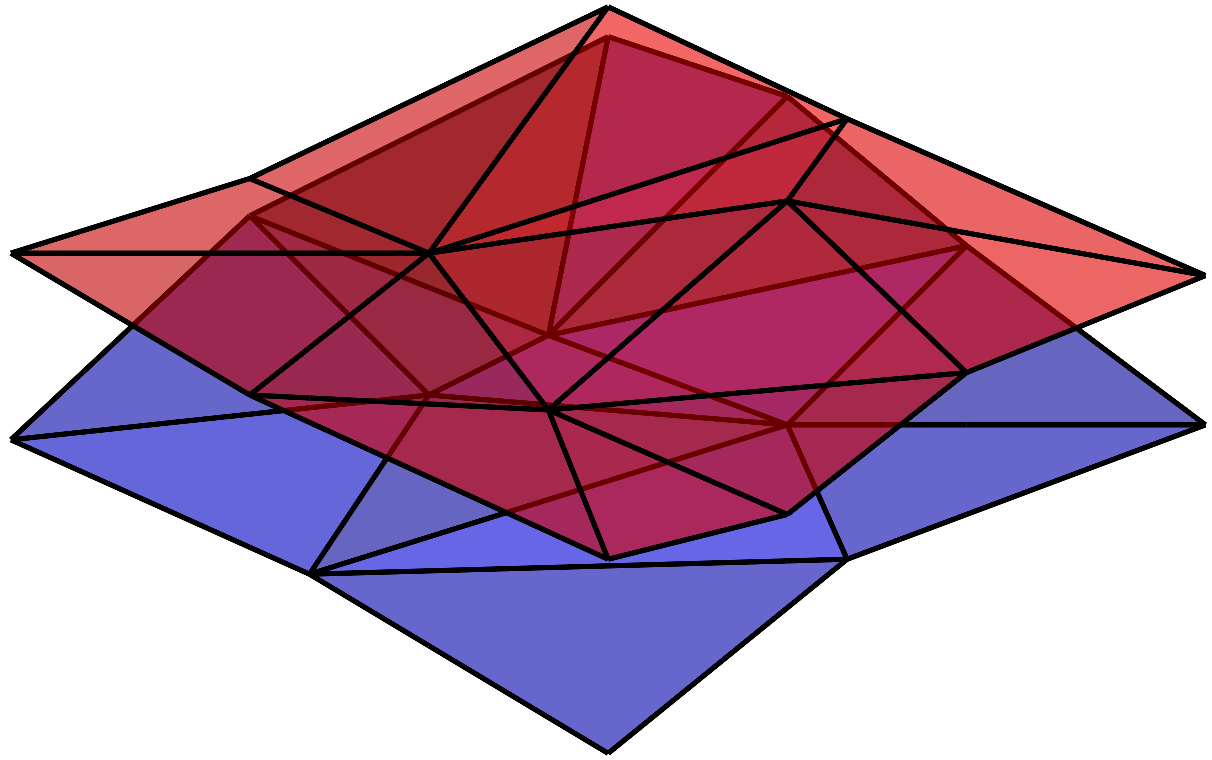
We want to find the translation and scaling of f to align it with g as good as possible.



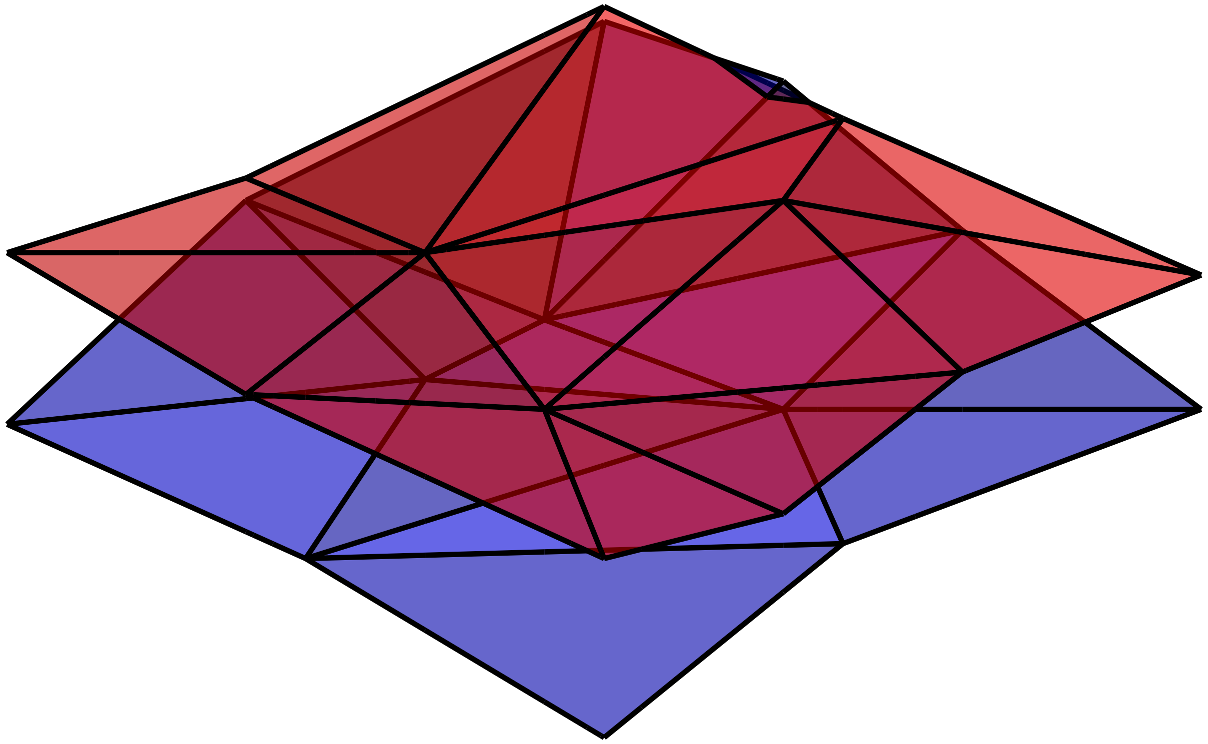
We want to find the translation and scaling of f to align it with g as good as possible.



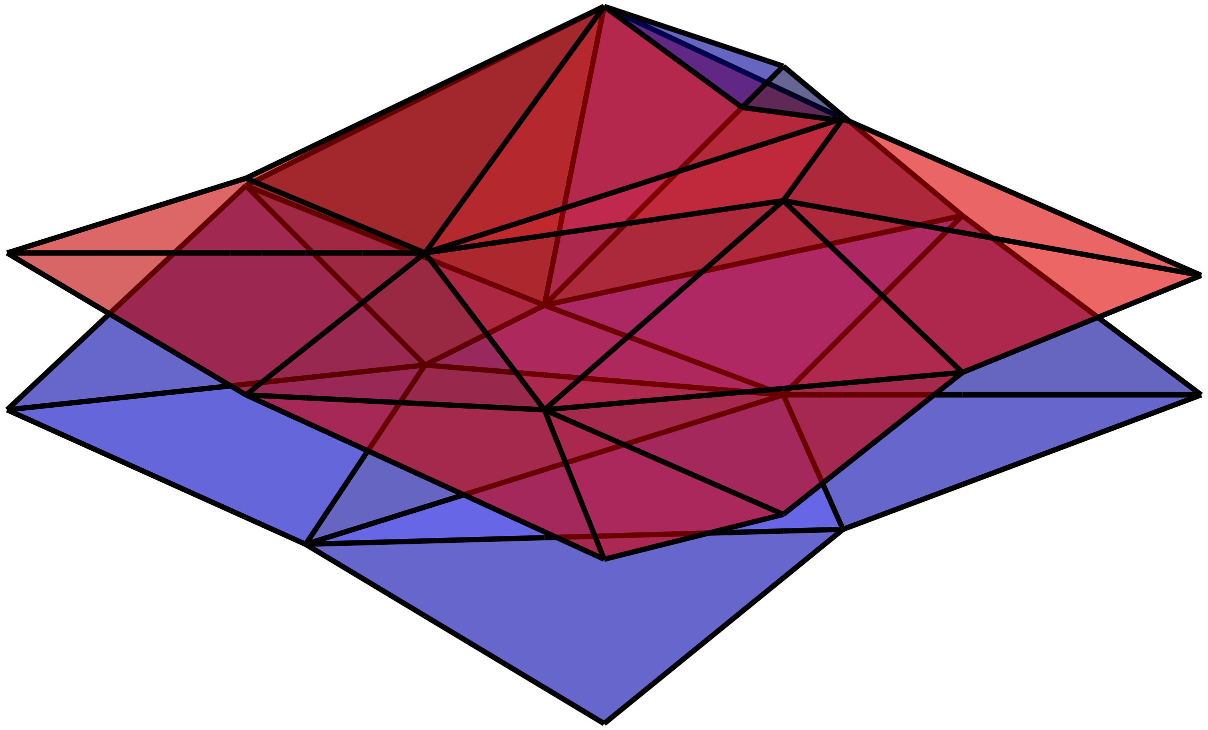
We want to find the translation and scaling of f to align it with g as good as possible.



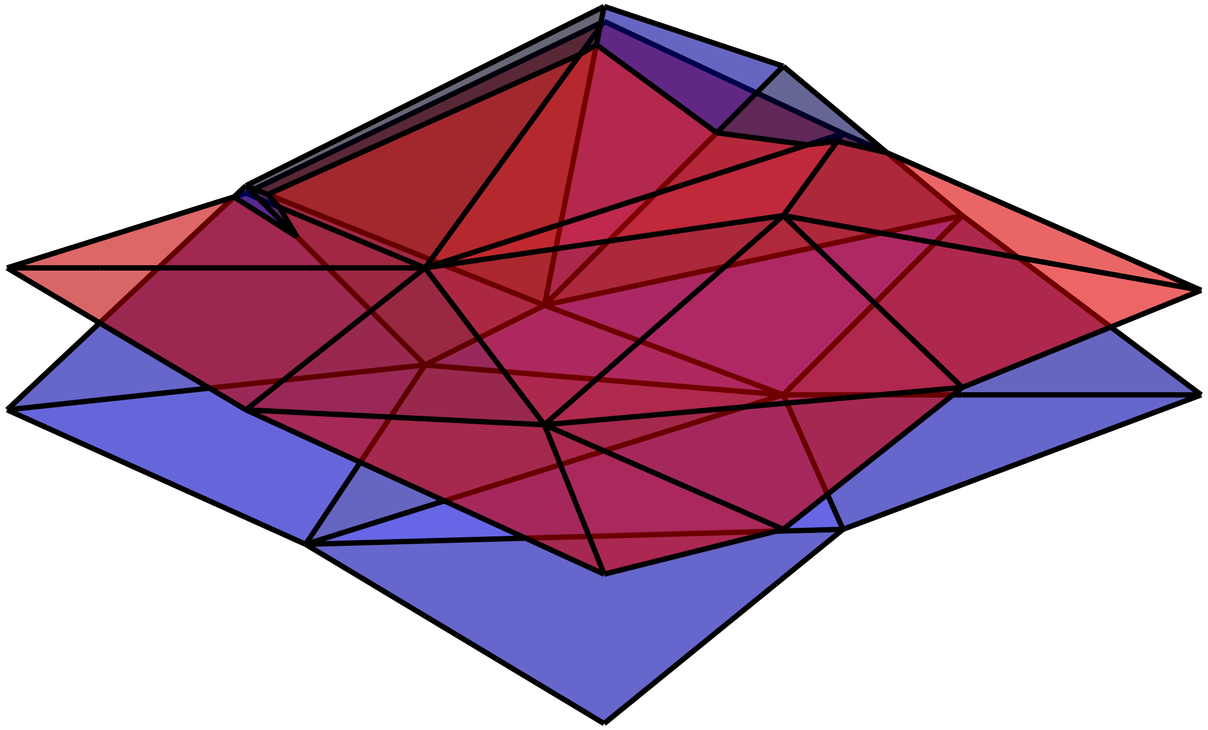
We want to find the translation and scaling of f to align it with g as good as possible.



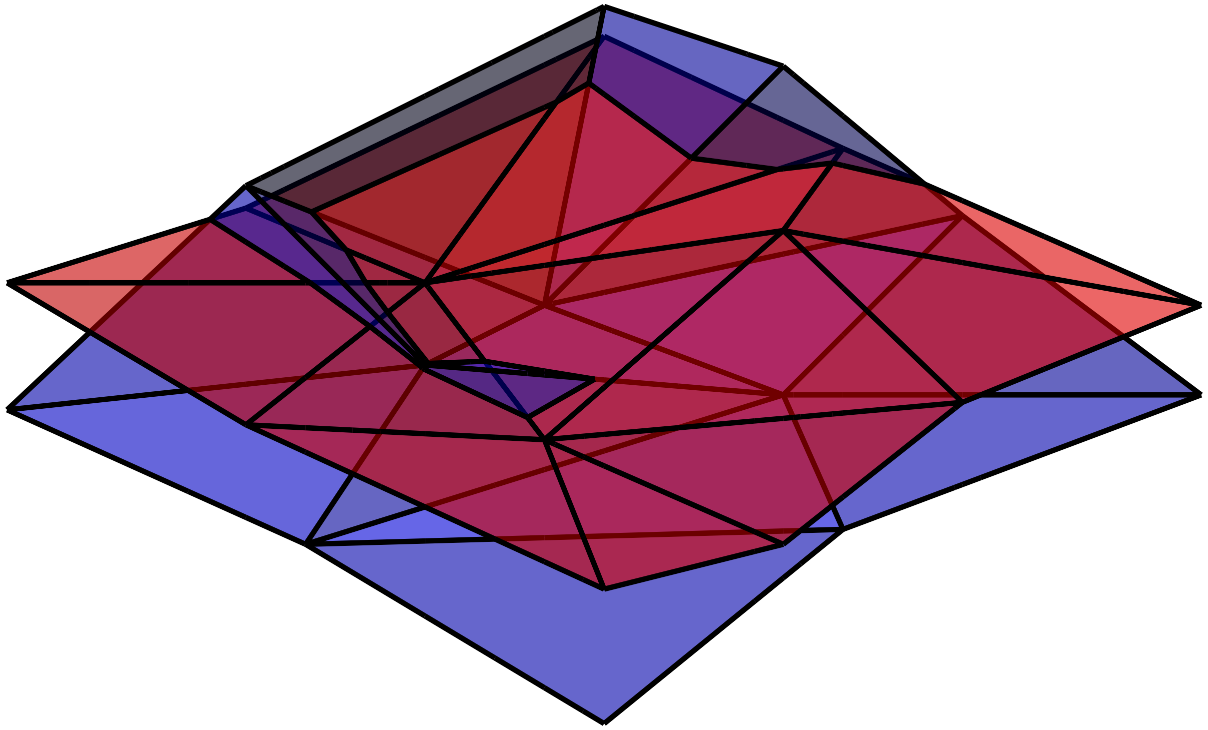
We want to find the translation and scaling of f to align it with g as good as possible.



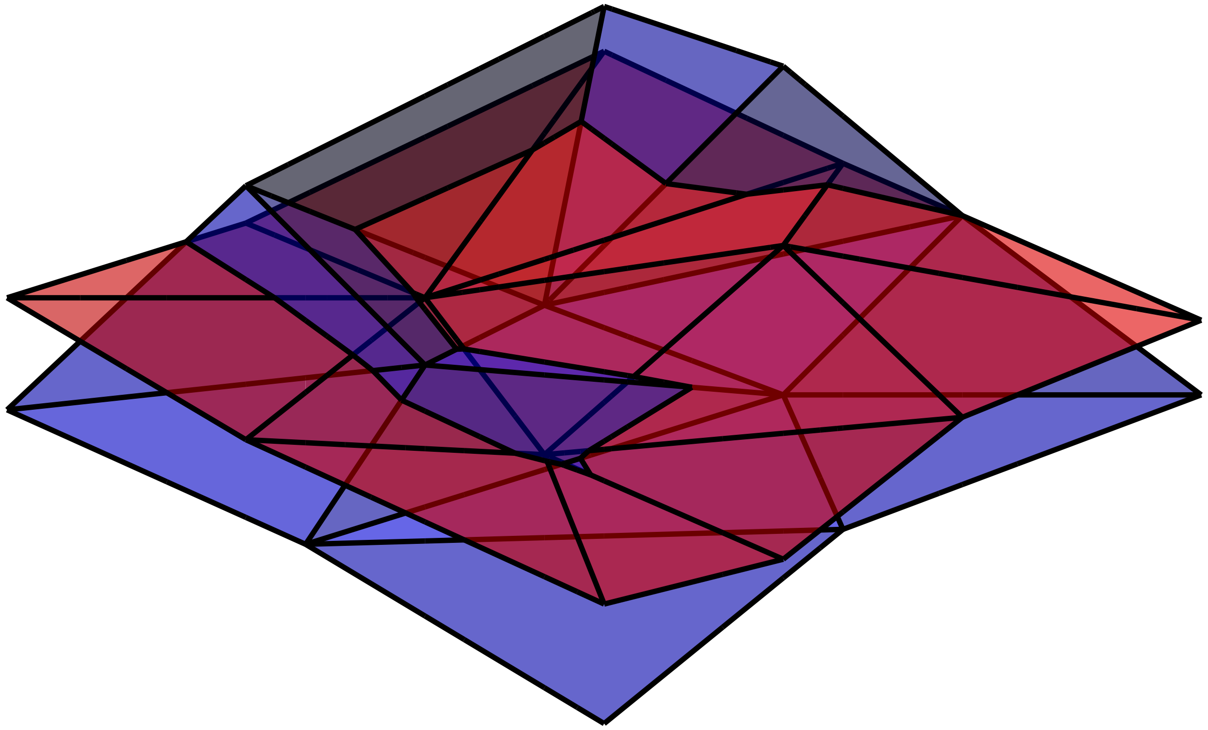
We want to find the translation and scaling of f to align it with g as good as possible.



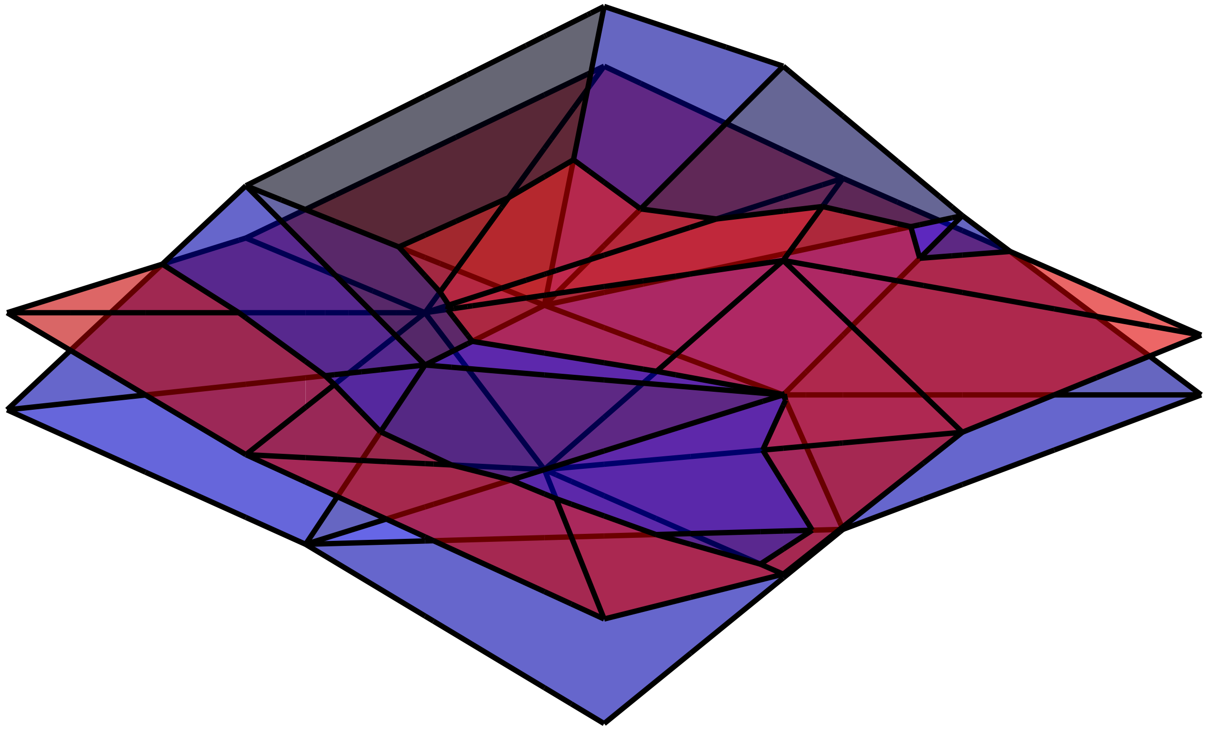
We want to find the translation and scaling of f to align it with g as good as possible.



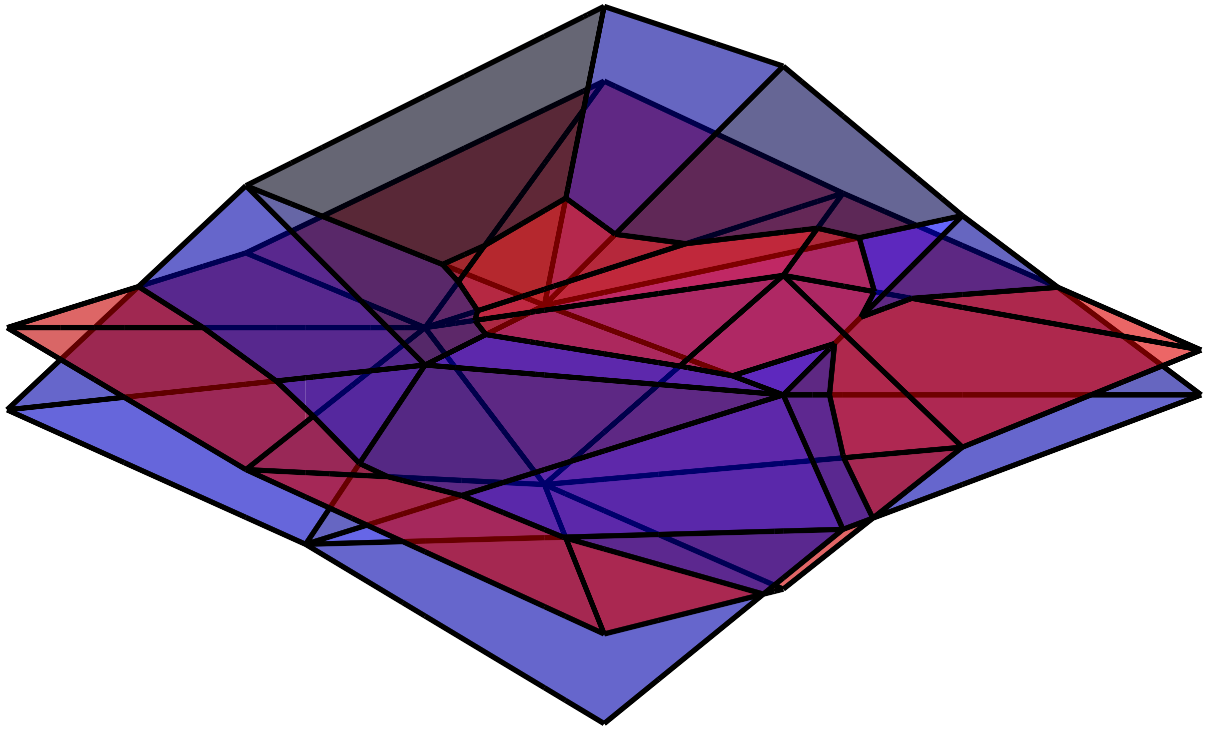
We want to find the translation and scaling of f to align it with g as good as possible.



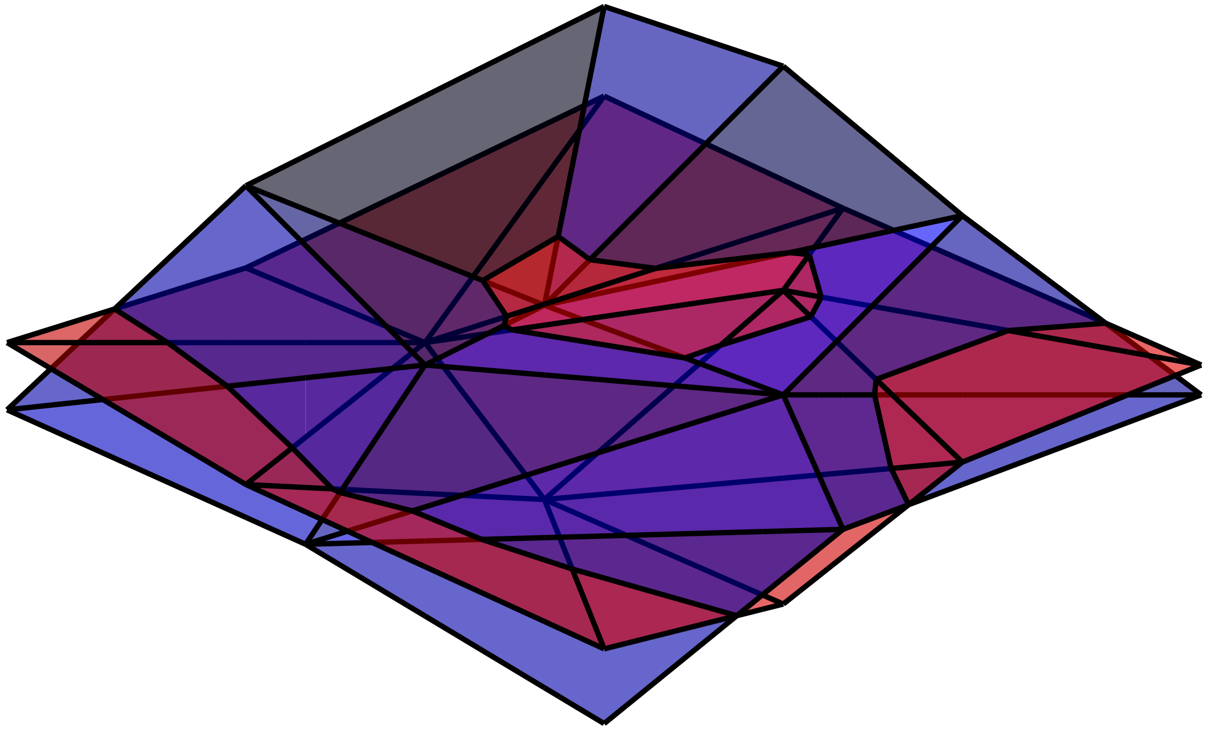
We want to find the translation and scaling of f to align it with g as good as possible.



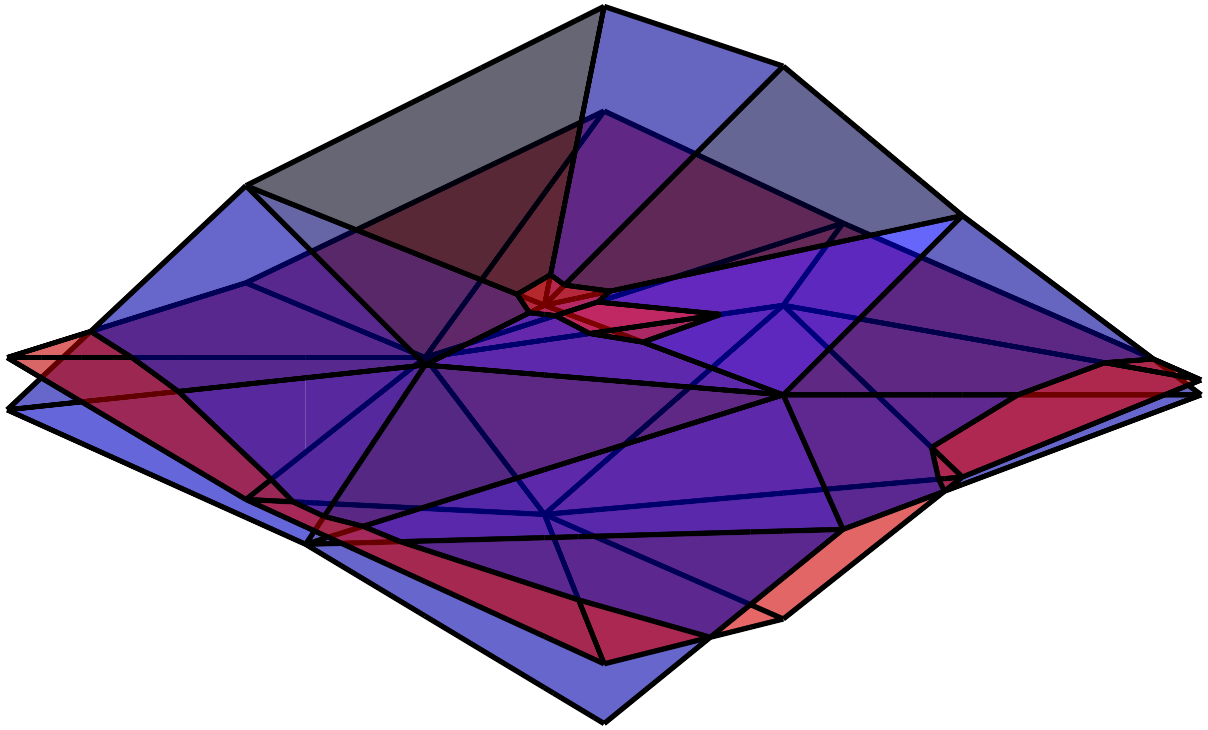
We want to find the translation and scaling of f to align it with g as good as possible.



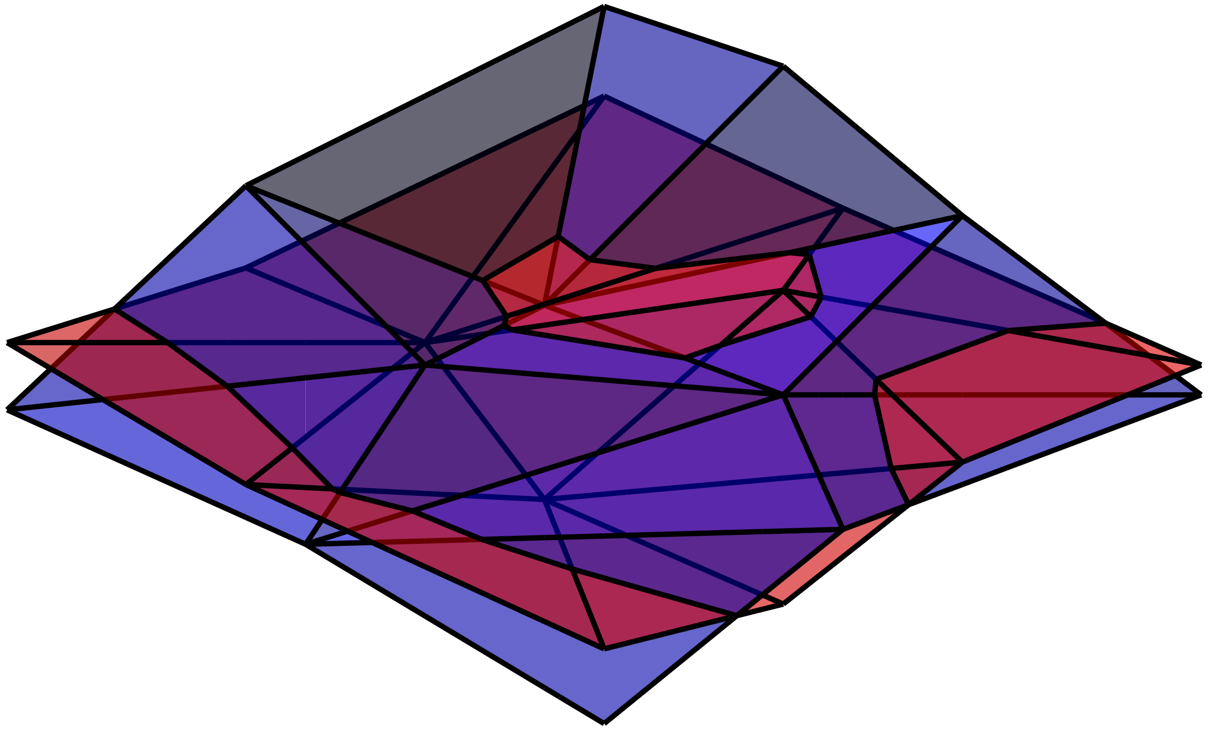
We want to find the translation and scaling of f to align it with g as good as possible.



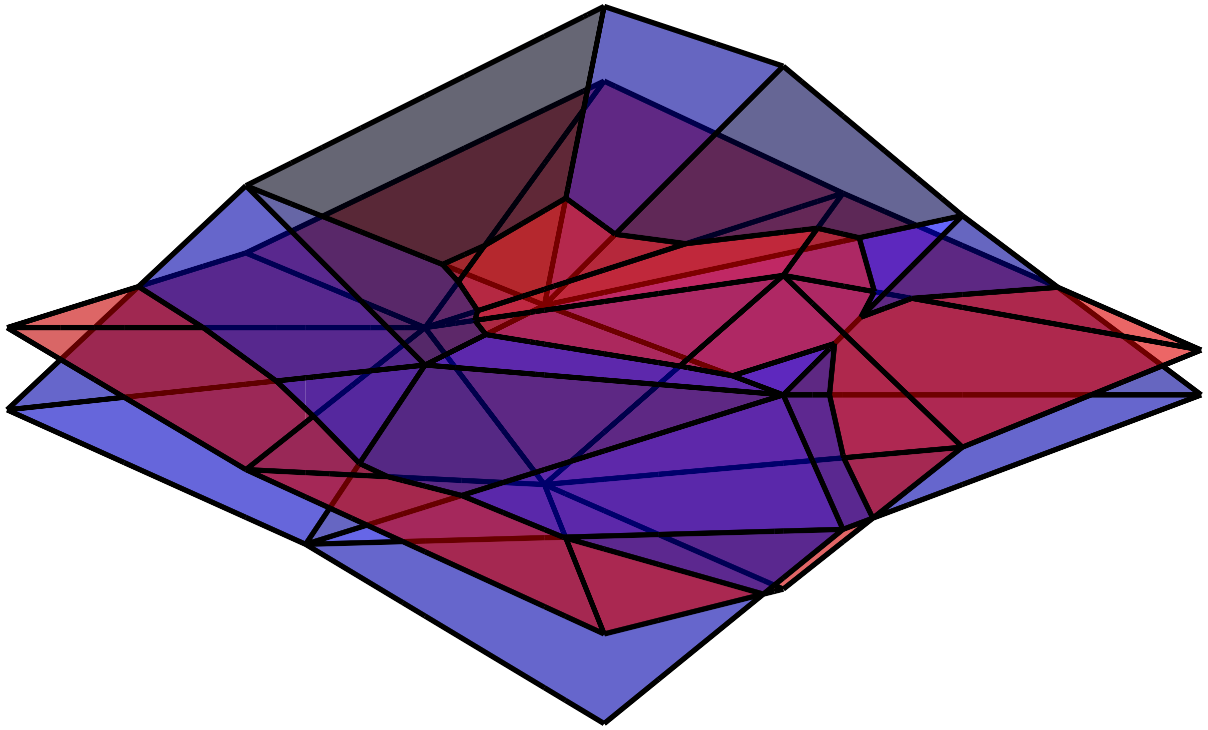
We want to find the translation and scaling of f to align it with g as good as possible.



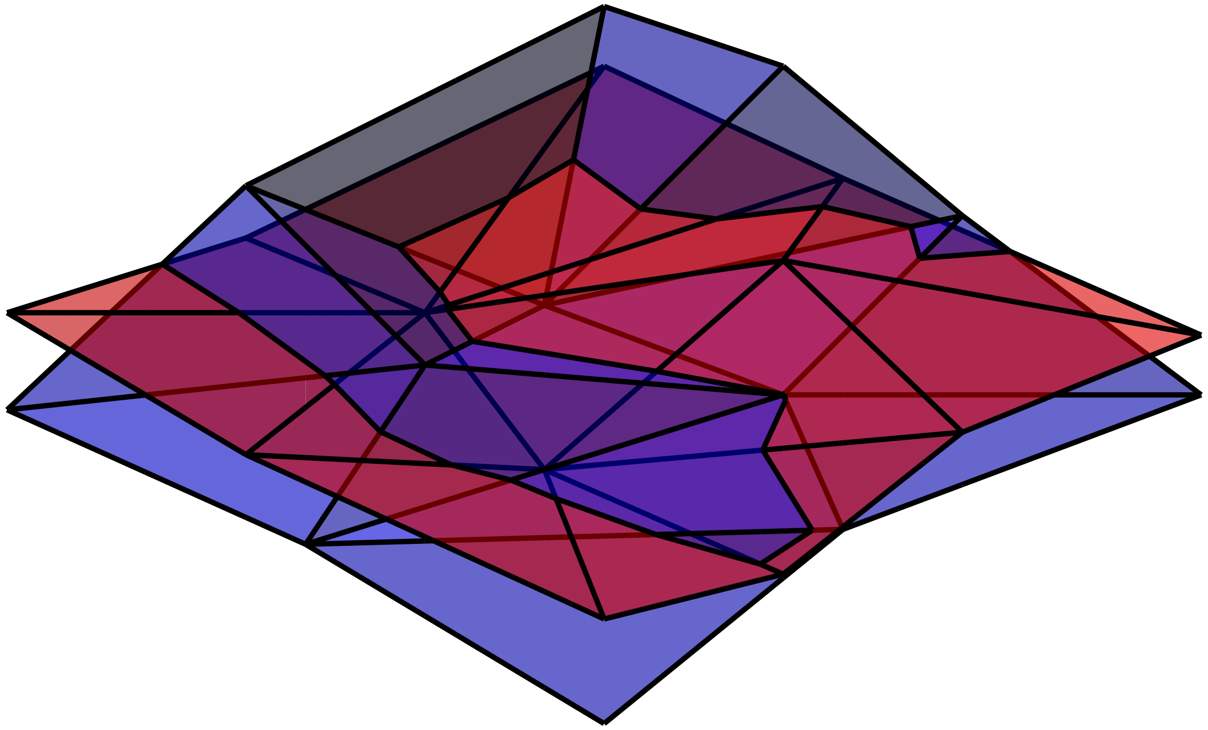
We want to find the translation and scaling of f to align it with g as good as possible.



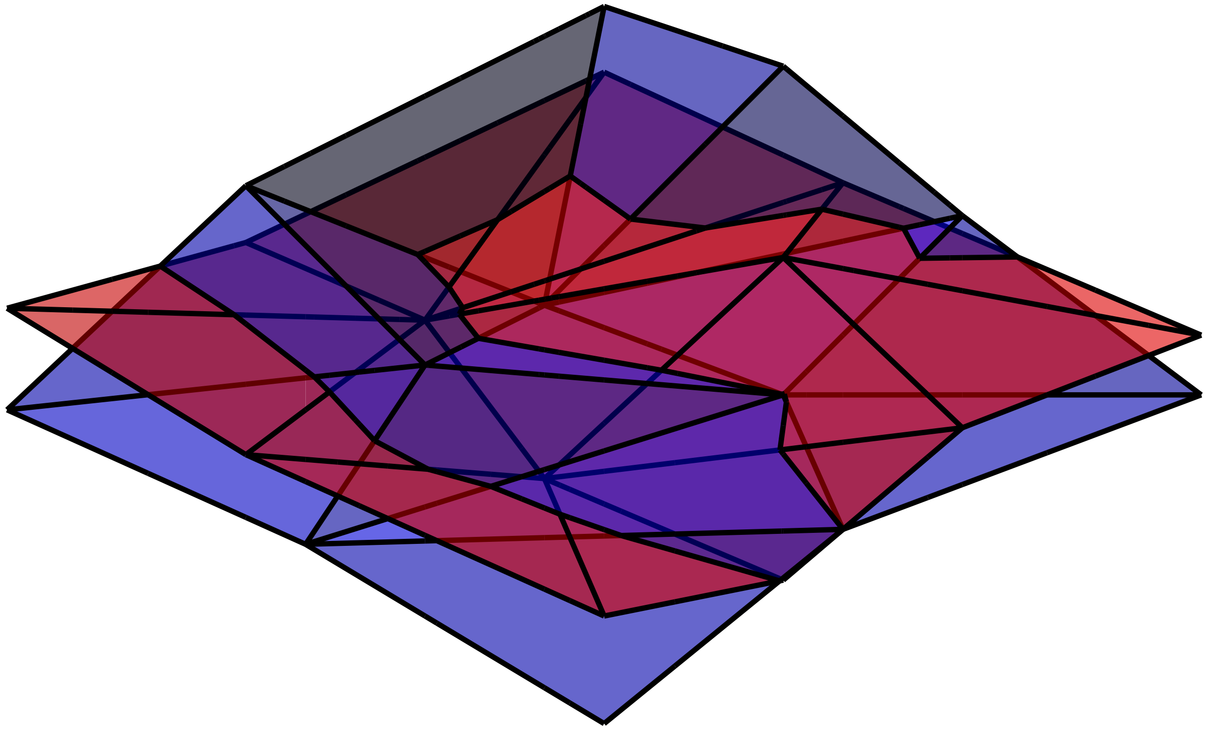
We want to find the translation and scaling of f to align it with g as good as possible.



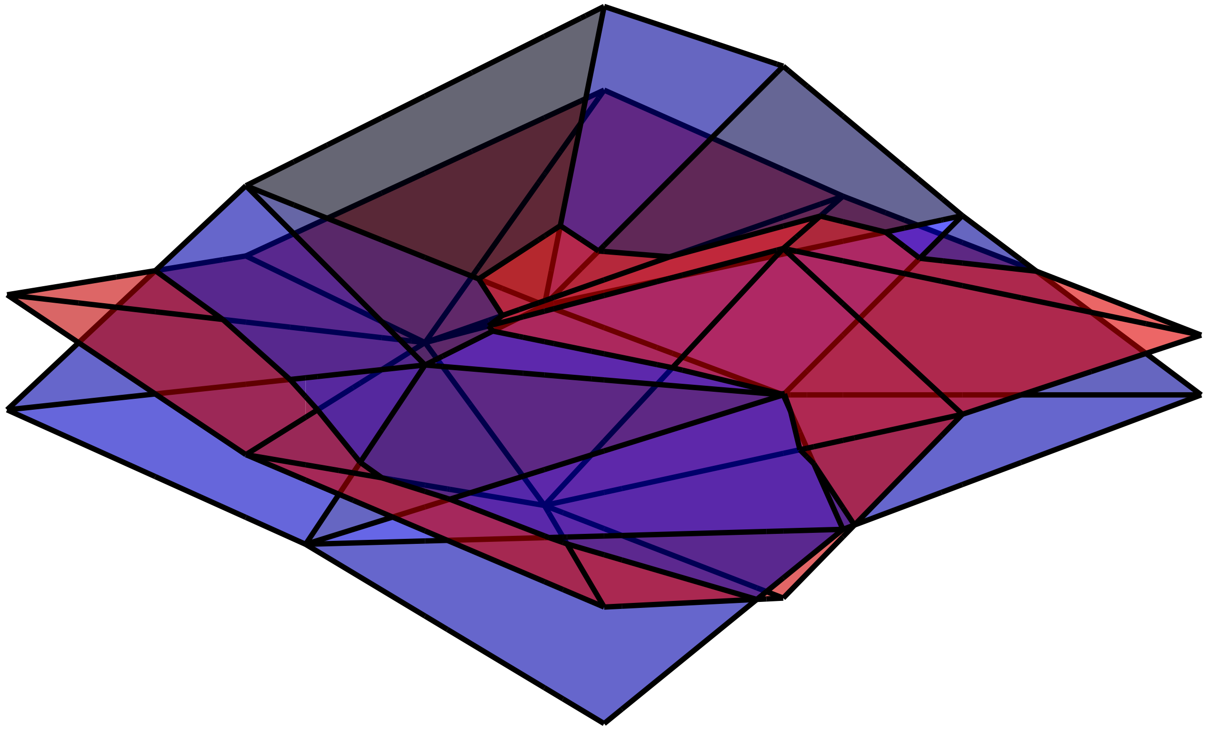
We want to find the translation and scaling of f to align it with g as good as possible.



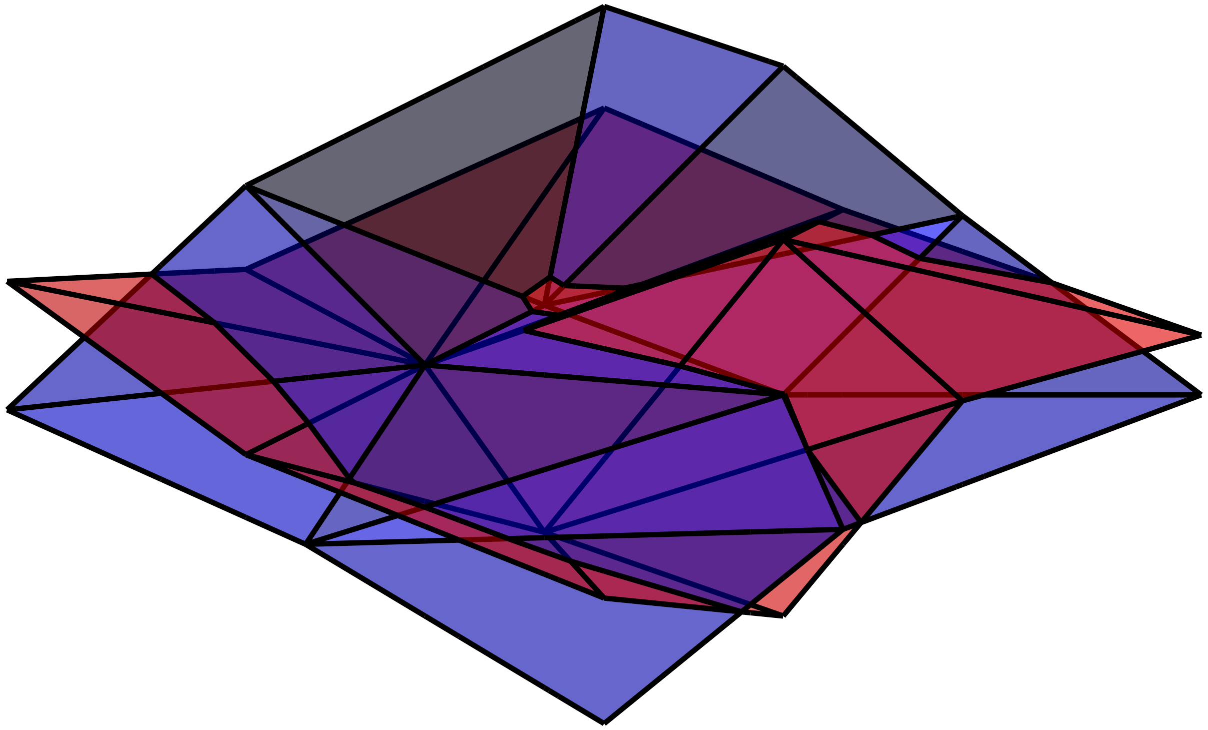
We want to find the translation and scaling of f to align it with g as good as possible.



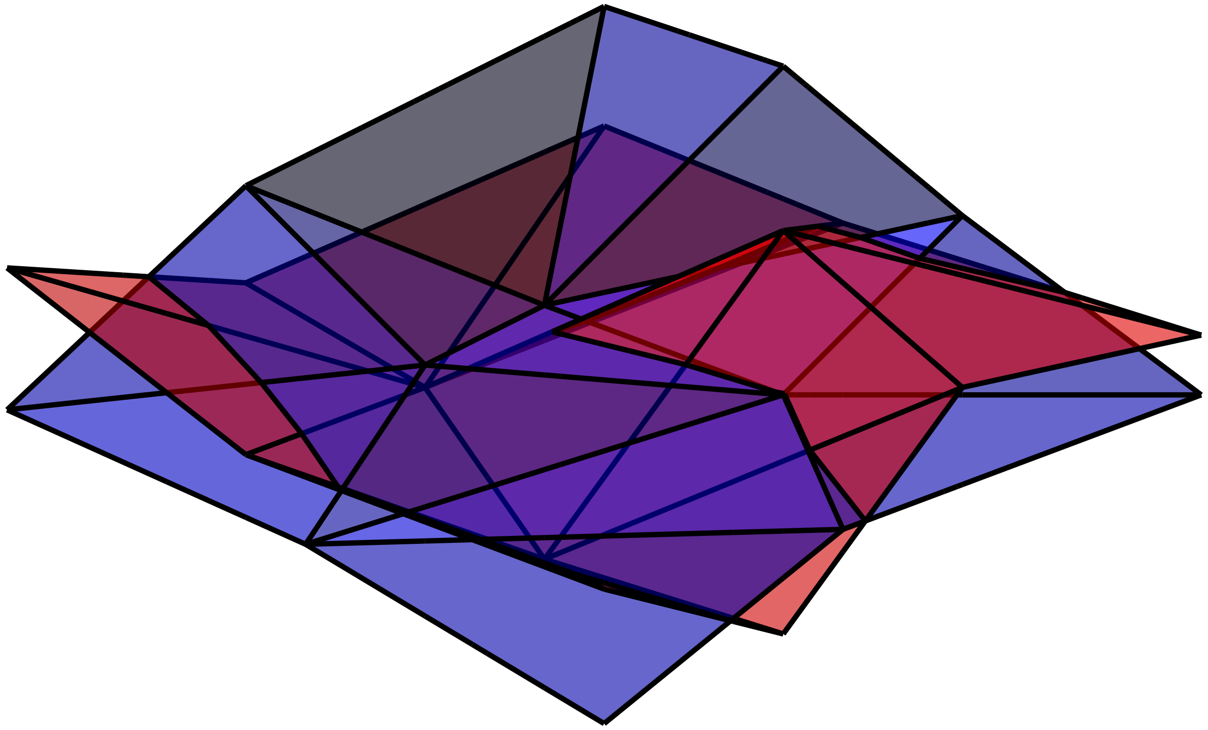
We want to find the translation and scaling of f to align it with g as good as possible.



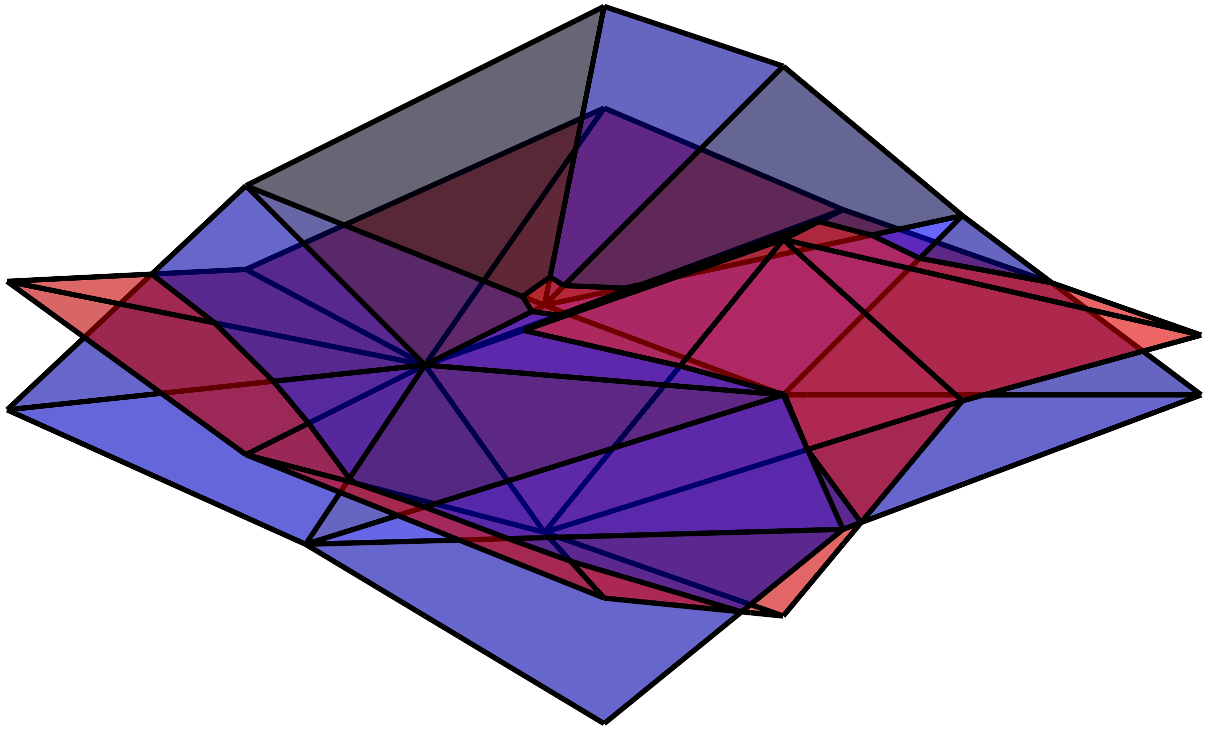
We want to find the translation and scaling of f to align it with g as good as possible.



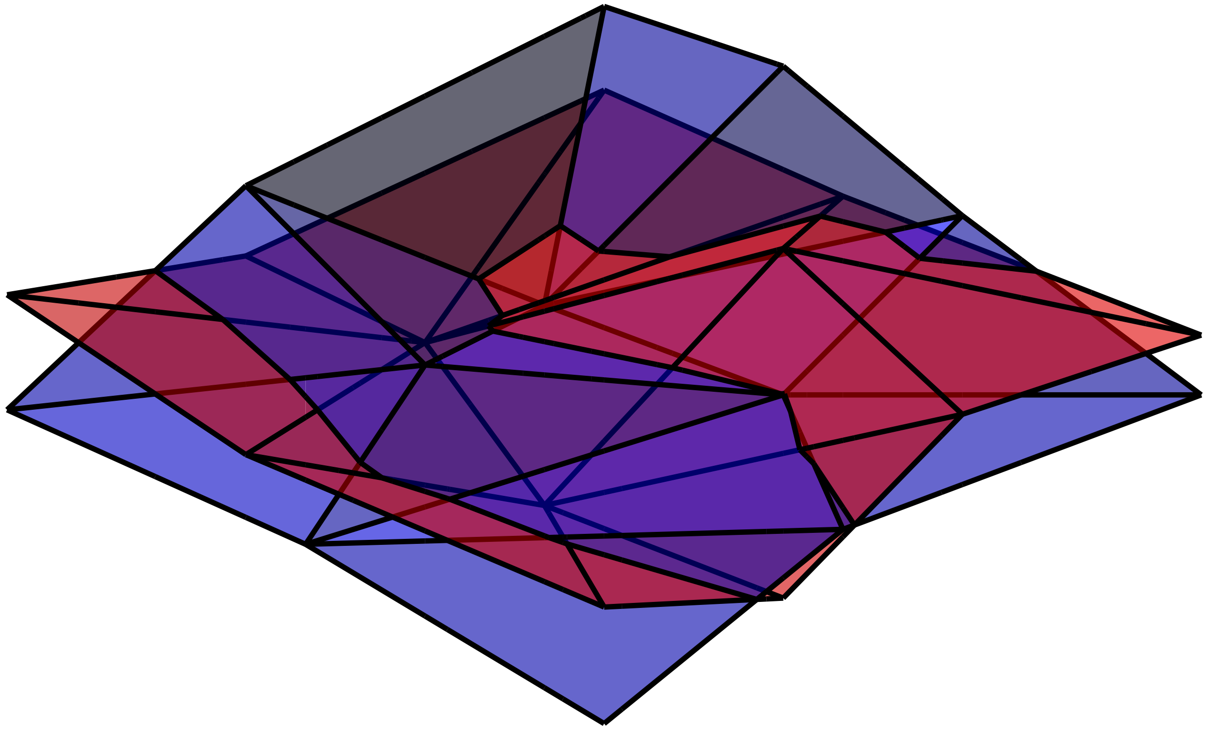
We want to find the translation and scaling of f to align it with g as good as possible.



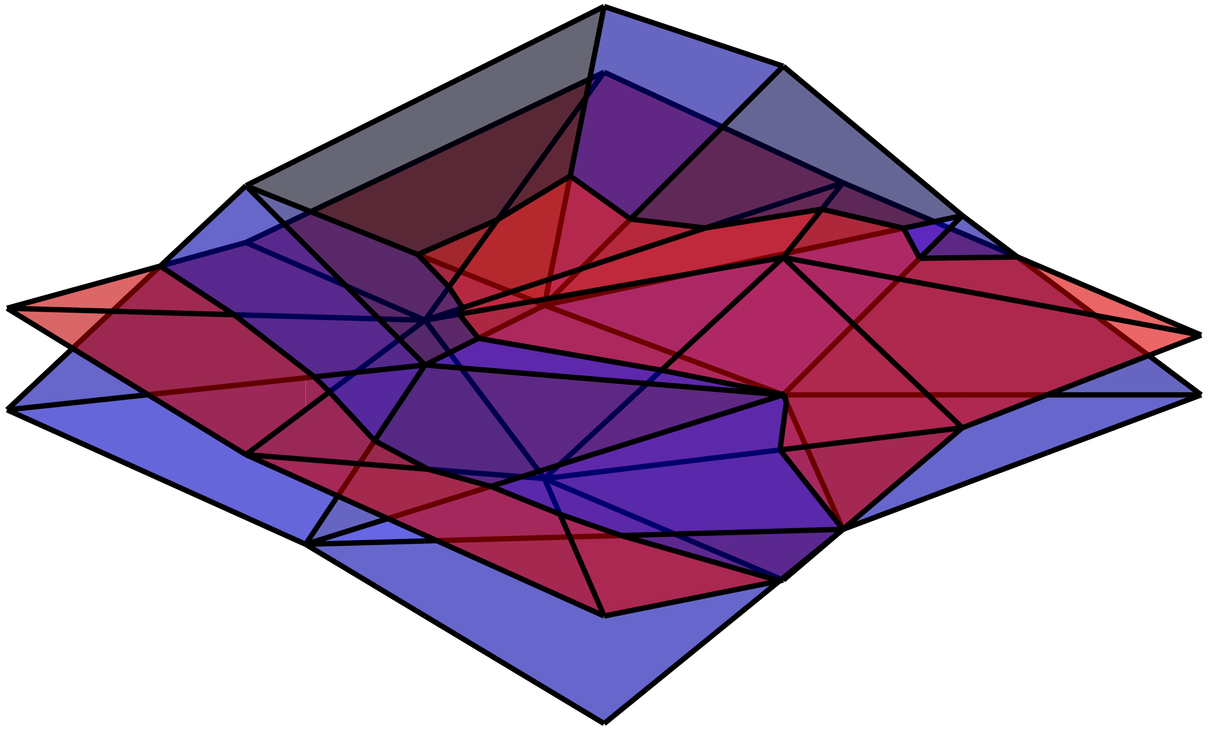
We want to find the translation and scaling of f to align it with g as good as possible.



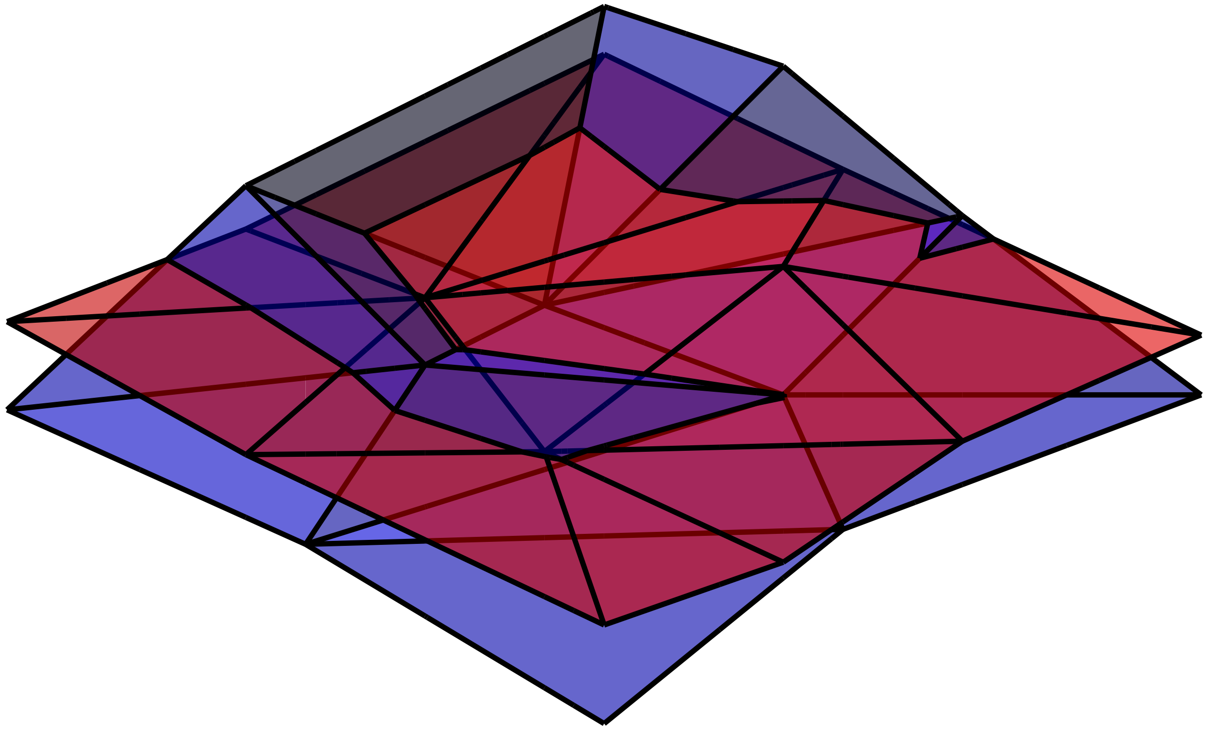
We want to find the translation and scaling of f to align it with g as good as possible.



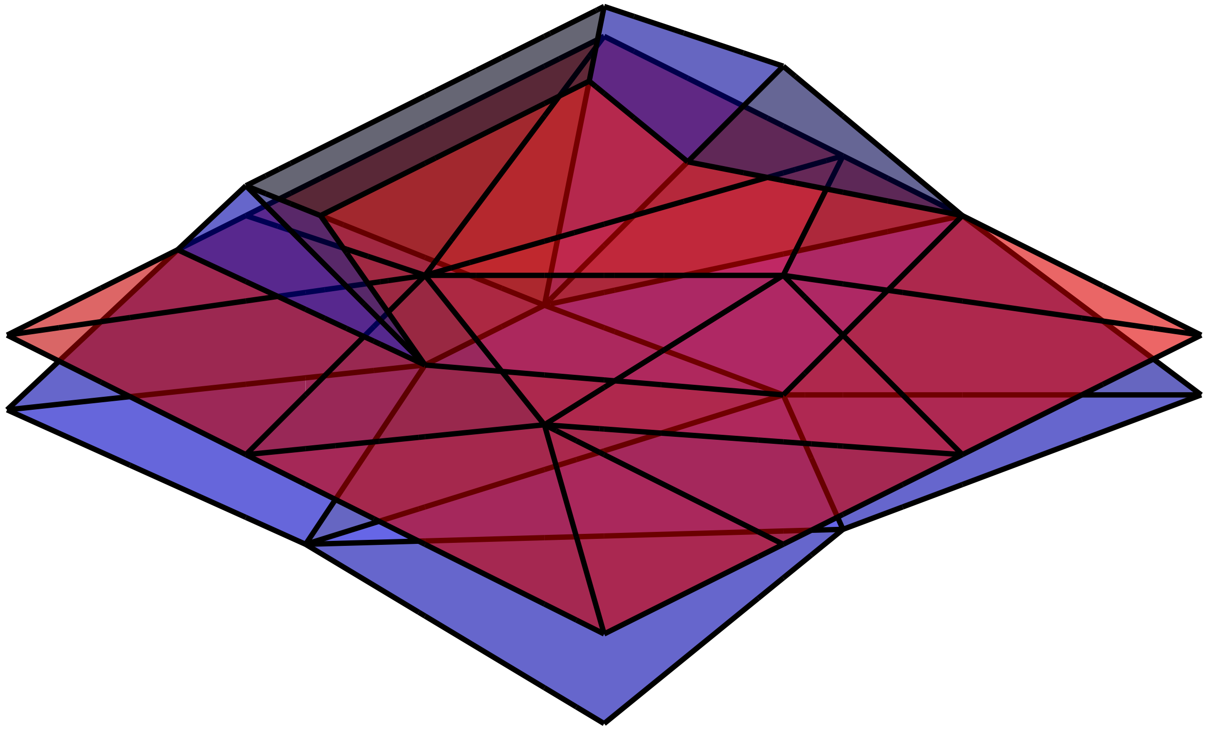
We want to find the translation and scaling of f to align it with g as good as possible.



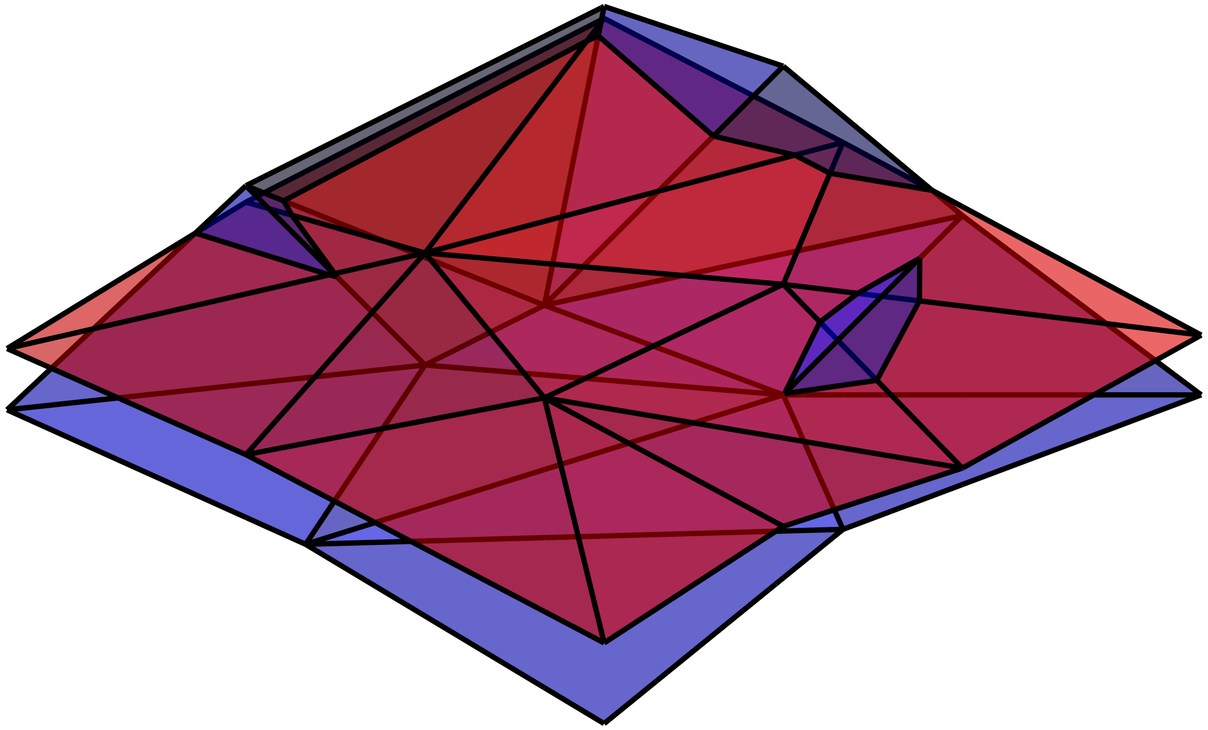
We want to find the translation and scaling of f to align it with g as good as possible.



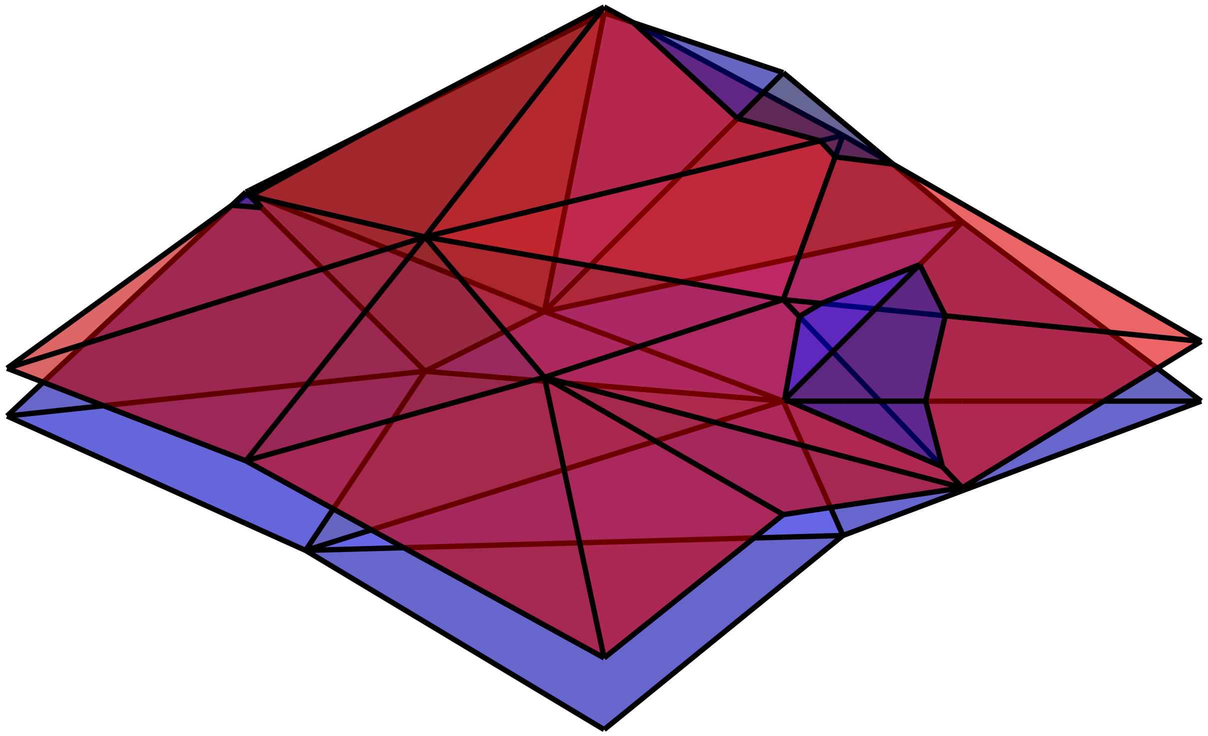
We want to find the translation and scaling of f to align it with g as good as possible.



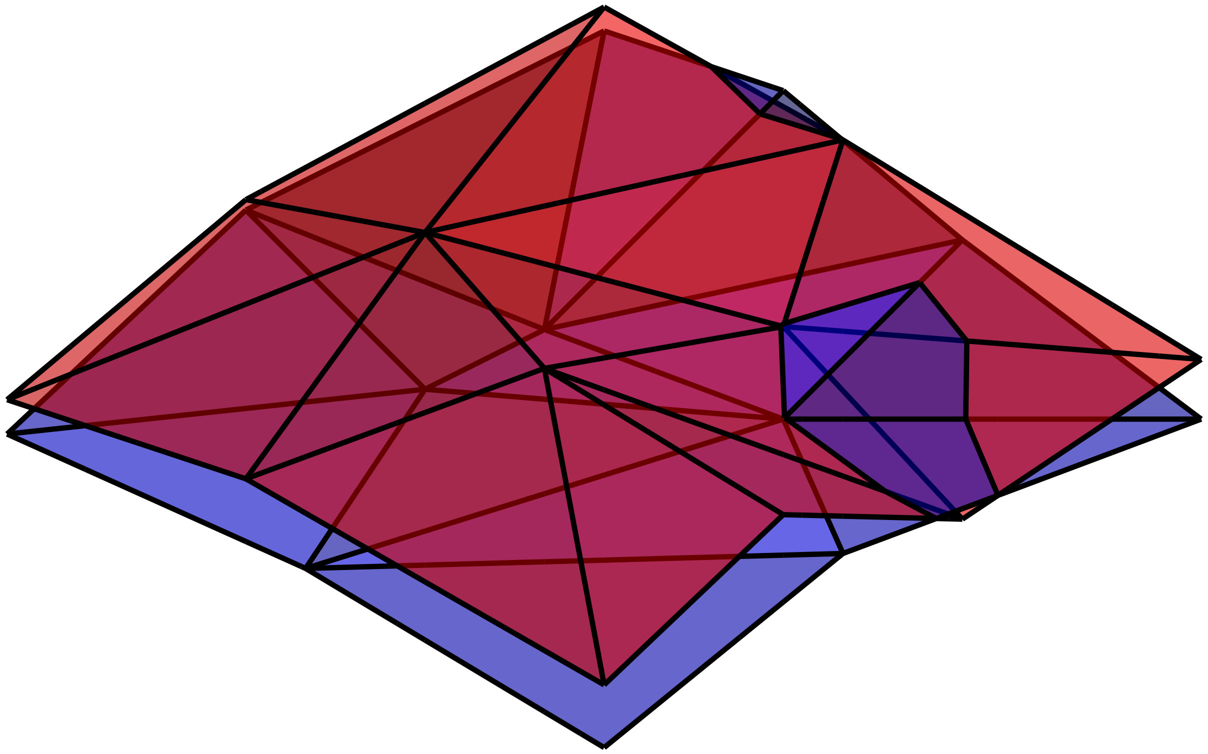
We want to find the translation and scaling of f to align it with g as good as possible.



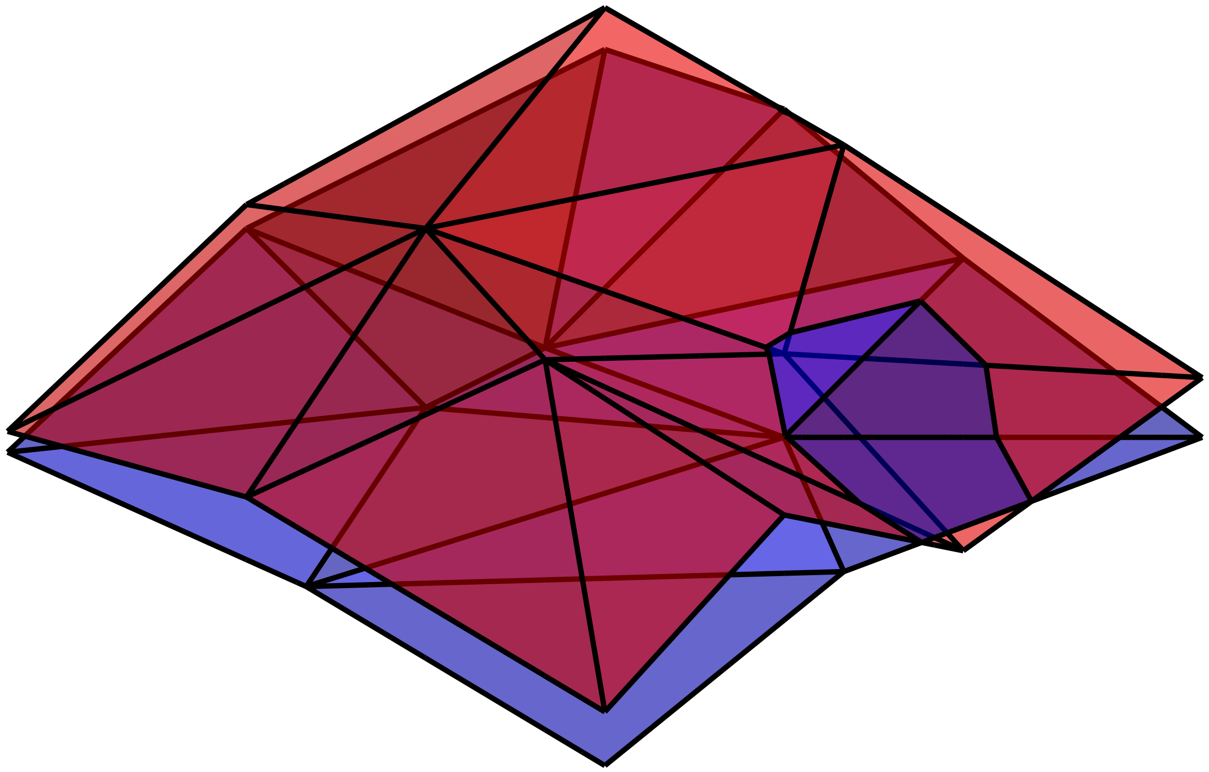
We want to find the translation and scaling of f to align it with g as good as possible.



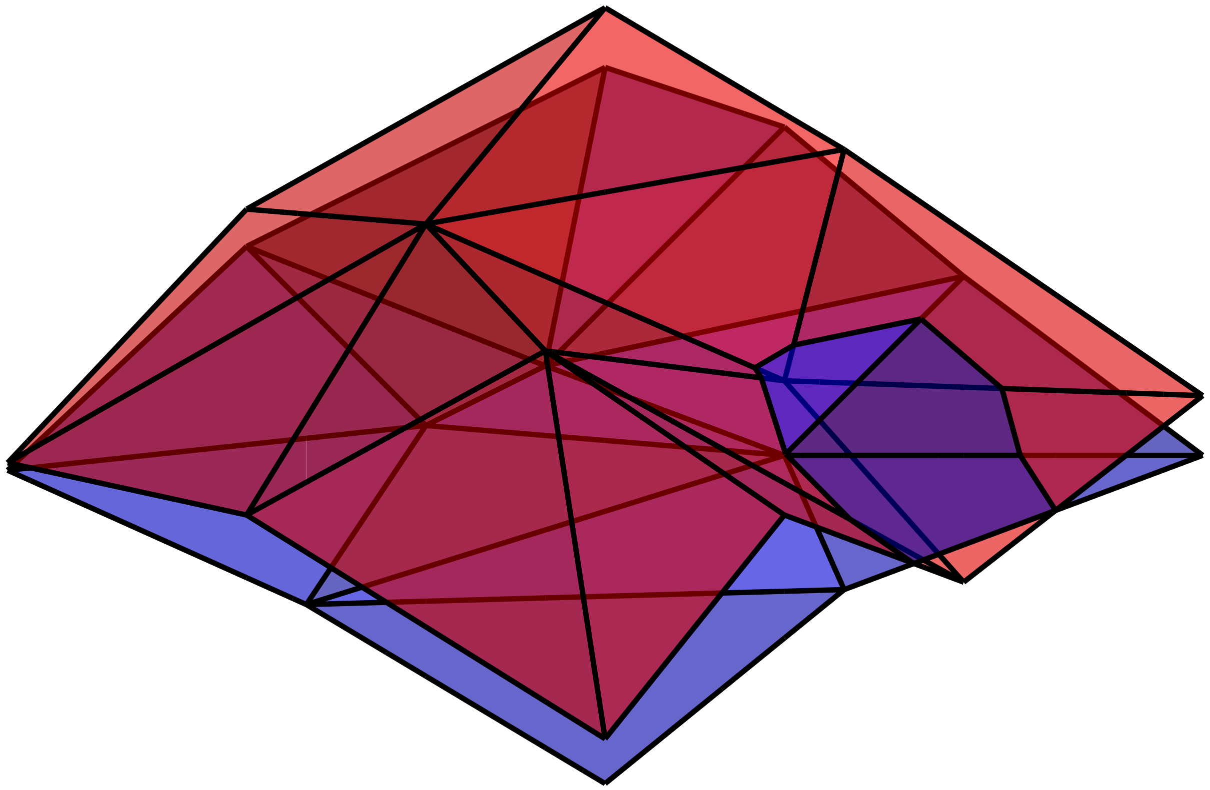
We want to find the translation and scaling of f to align it with g as good as possible.



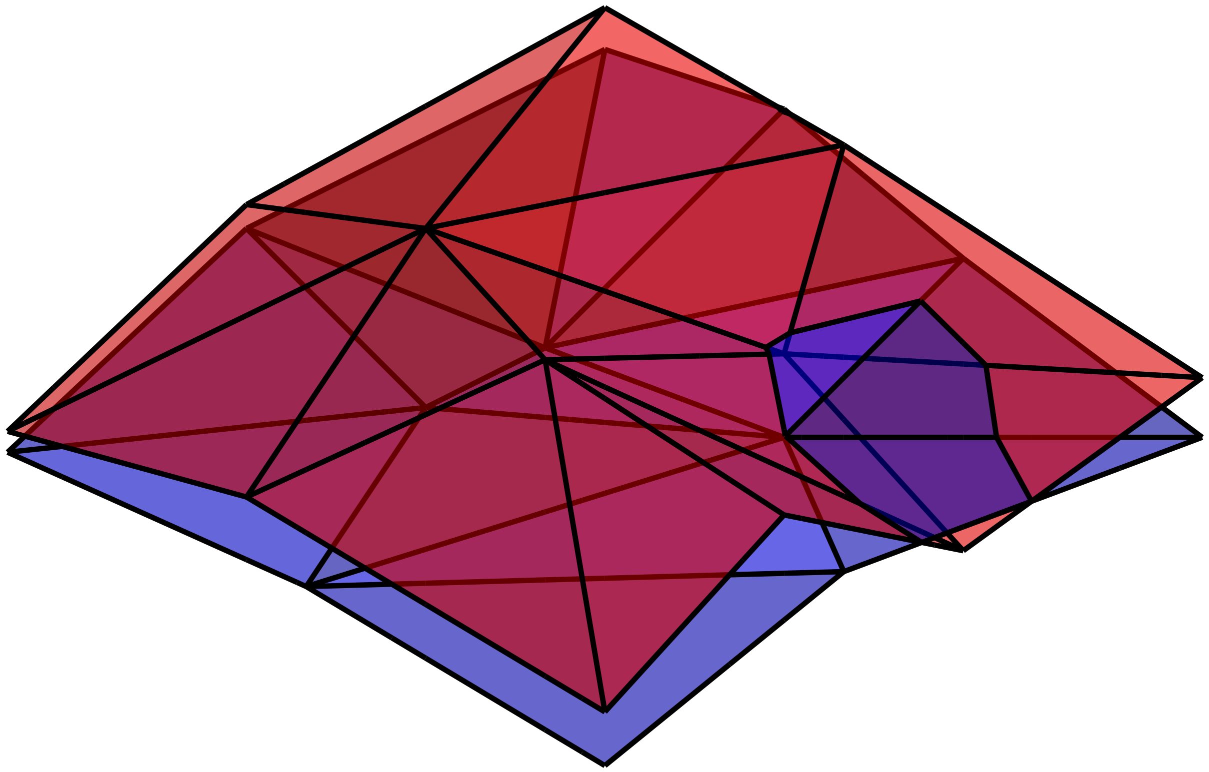
We want to find the translation and scaling of f to align it with g as good as possible.



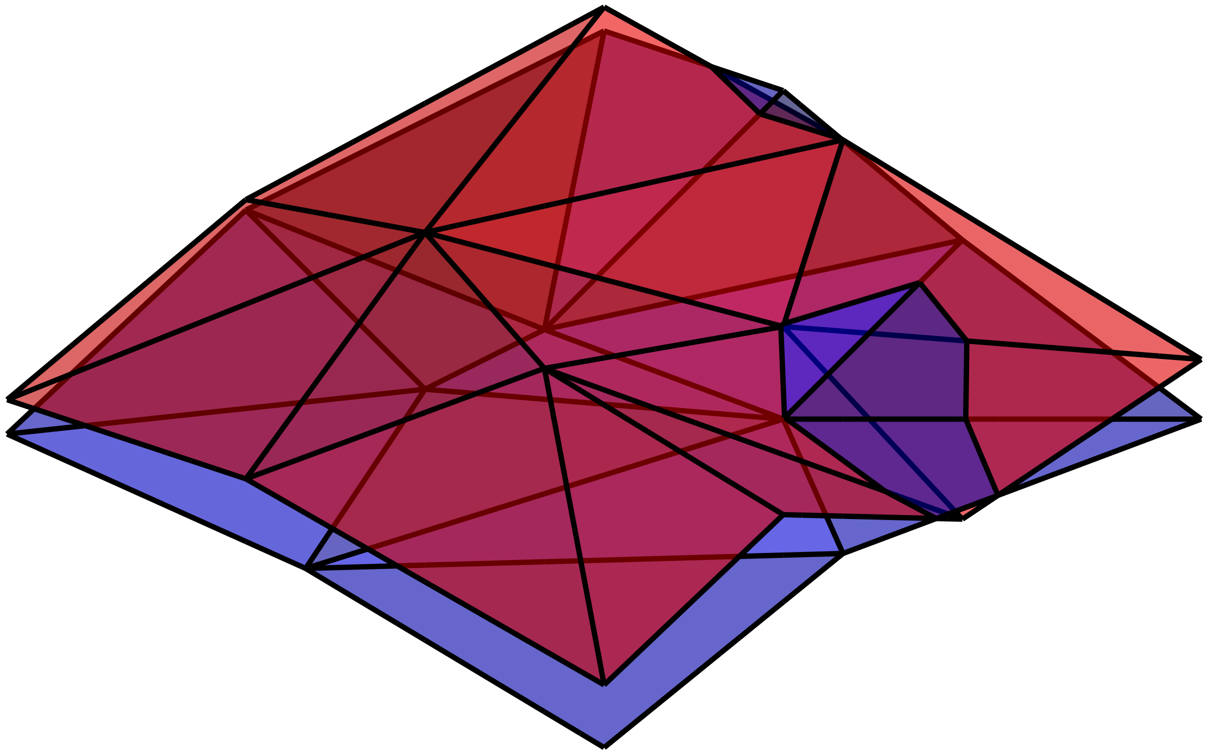
We want to find the translation and scaling of f to align it with g as good as possible.



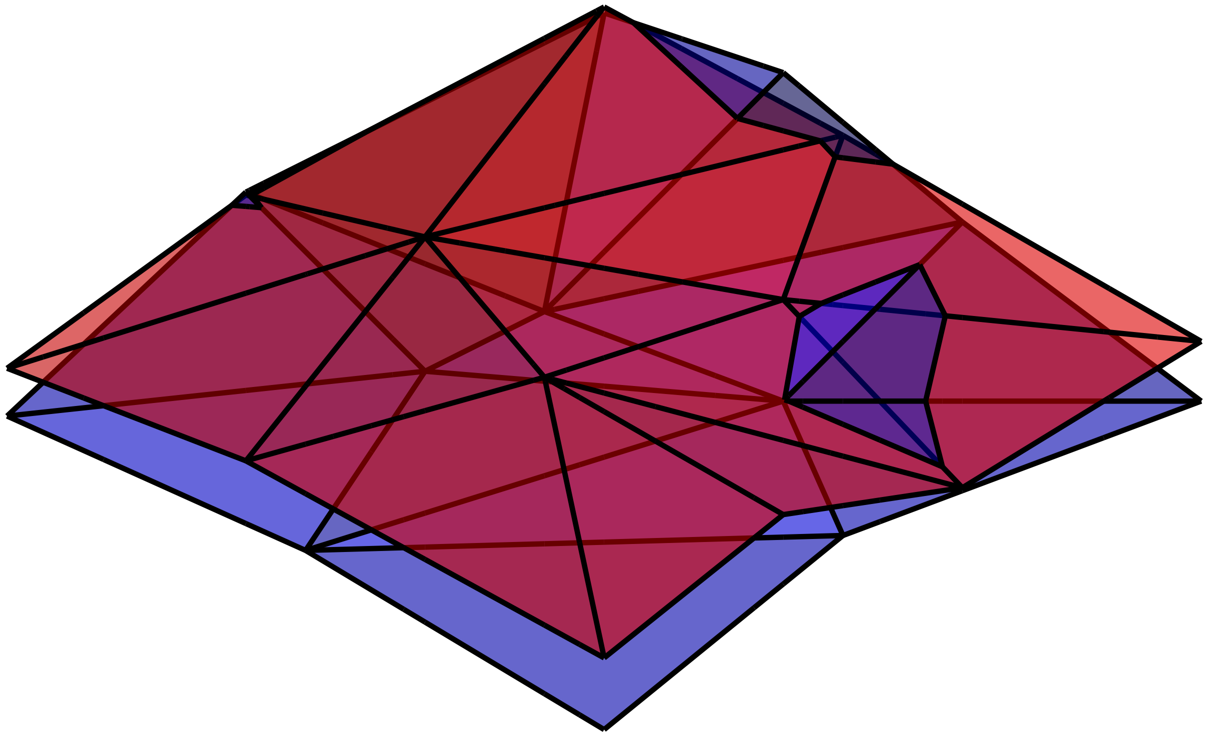
We want to find the translation and scaling of f to align it with g as good as possible.



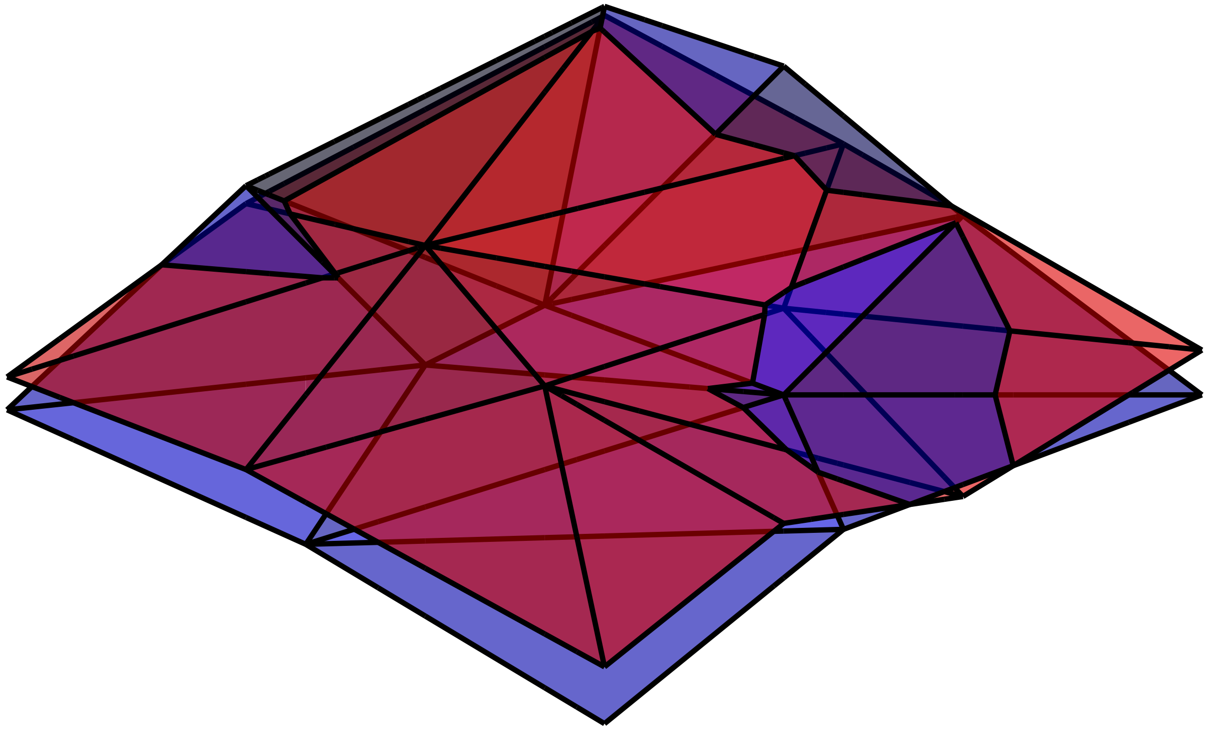
We want to find the translation and scaling of f to align it with g as good as possible.



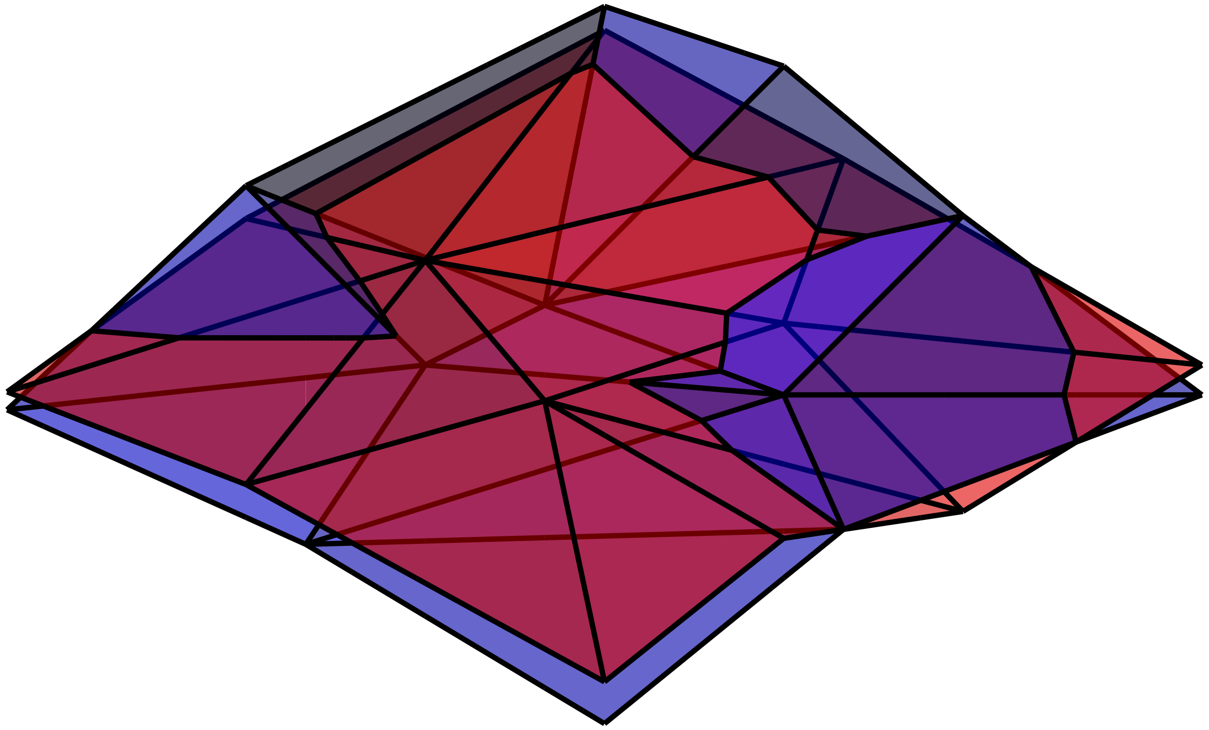
We want to find the translation and scaling of f to align it with g as good as possible.



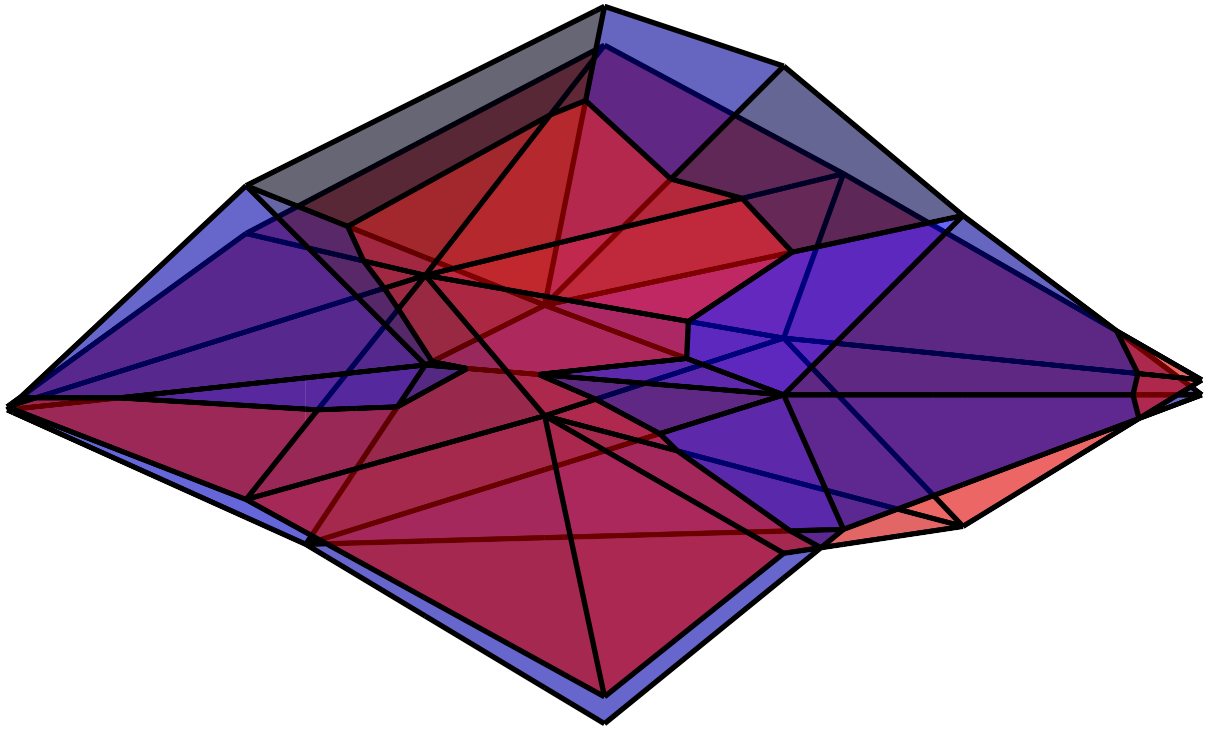
We want to find the translation and scaling of f to align it with g as good as possible.



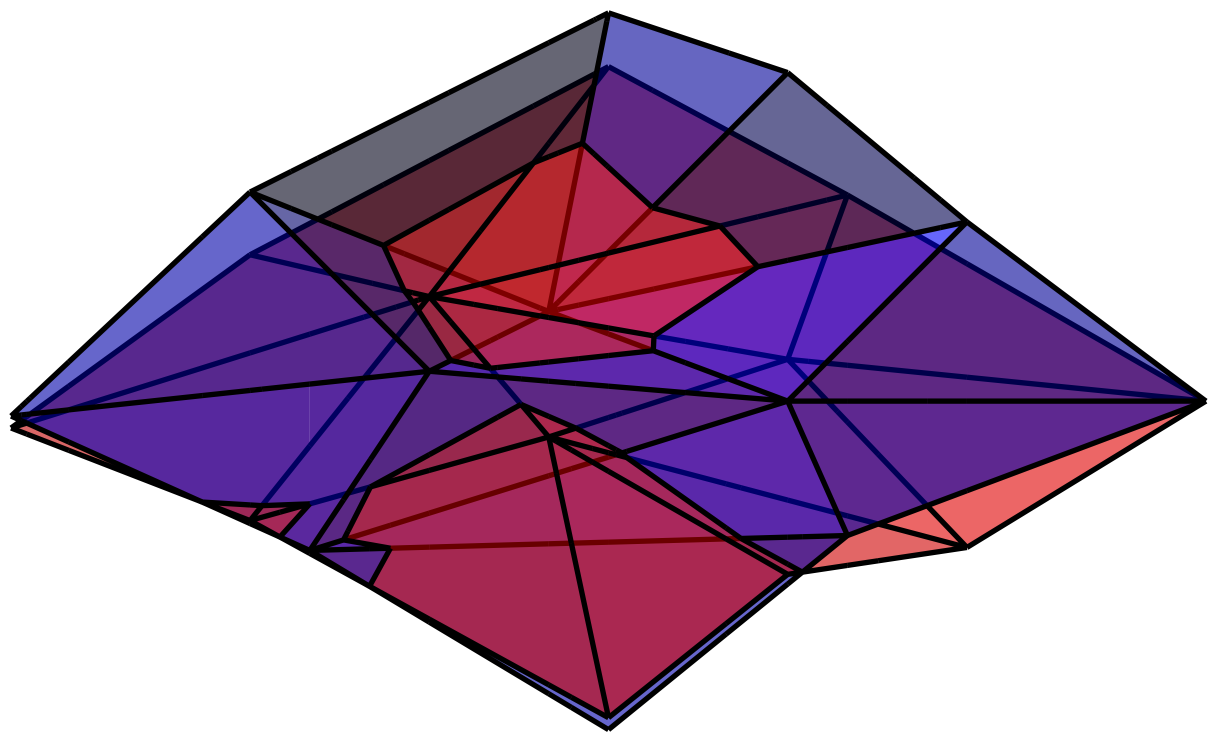
We want to find the translation and scaling of f to align it with g as good as possible.



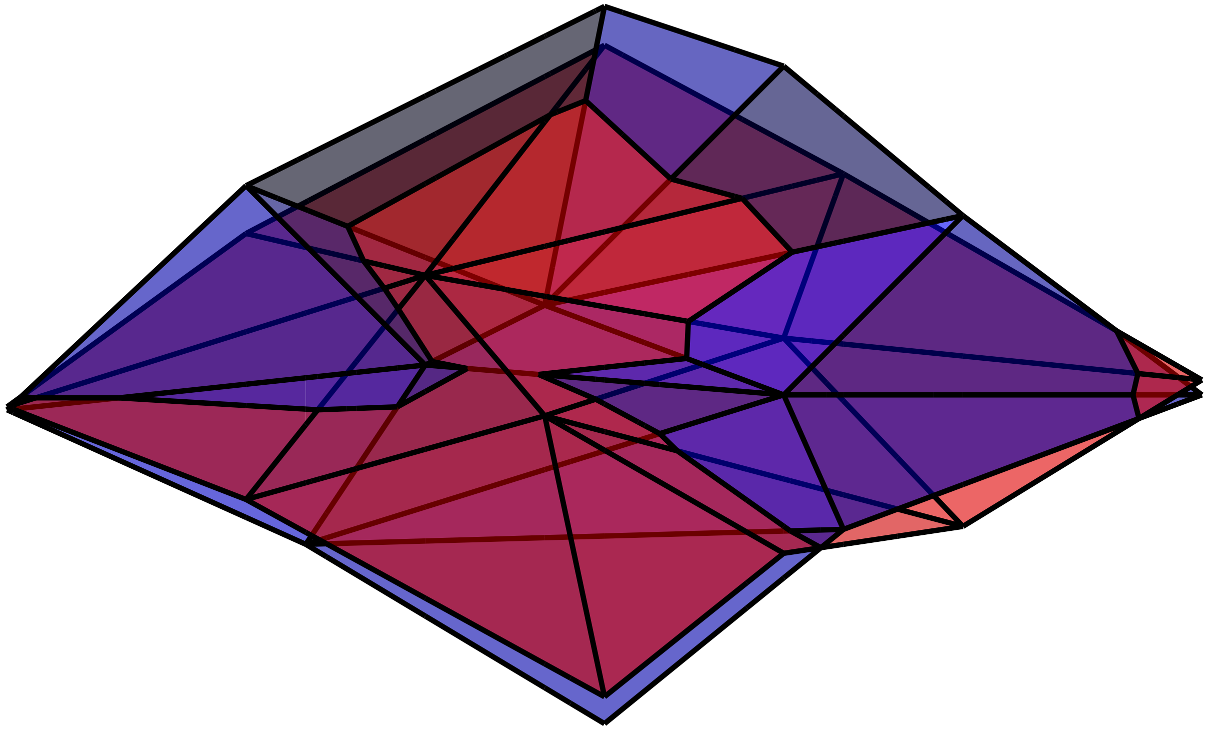
We want to find the translation and scaling of f to align it with g as good as possible.



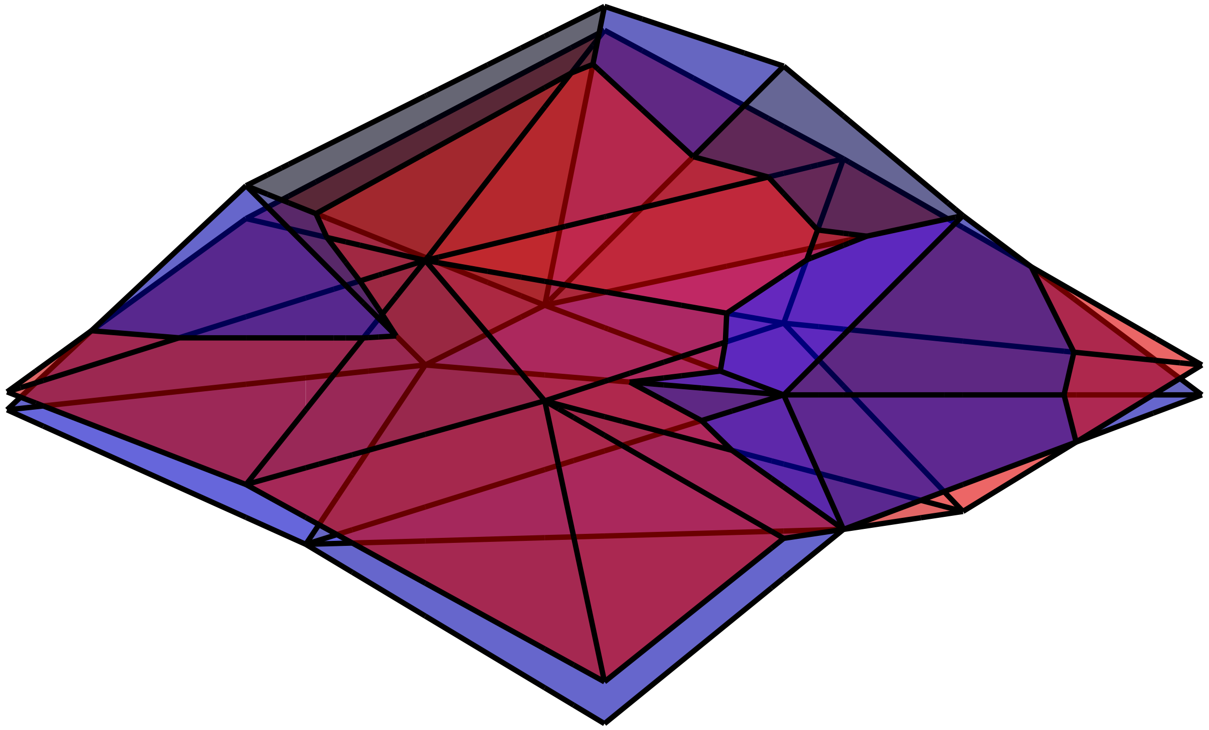
We want to find the translation and scaling of f to align it with g as good as possible.



We want to find the translation and scaling of f to align it with g as good as possible.

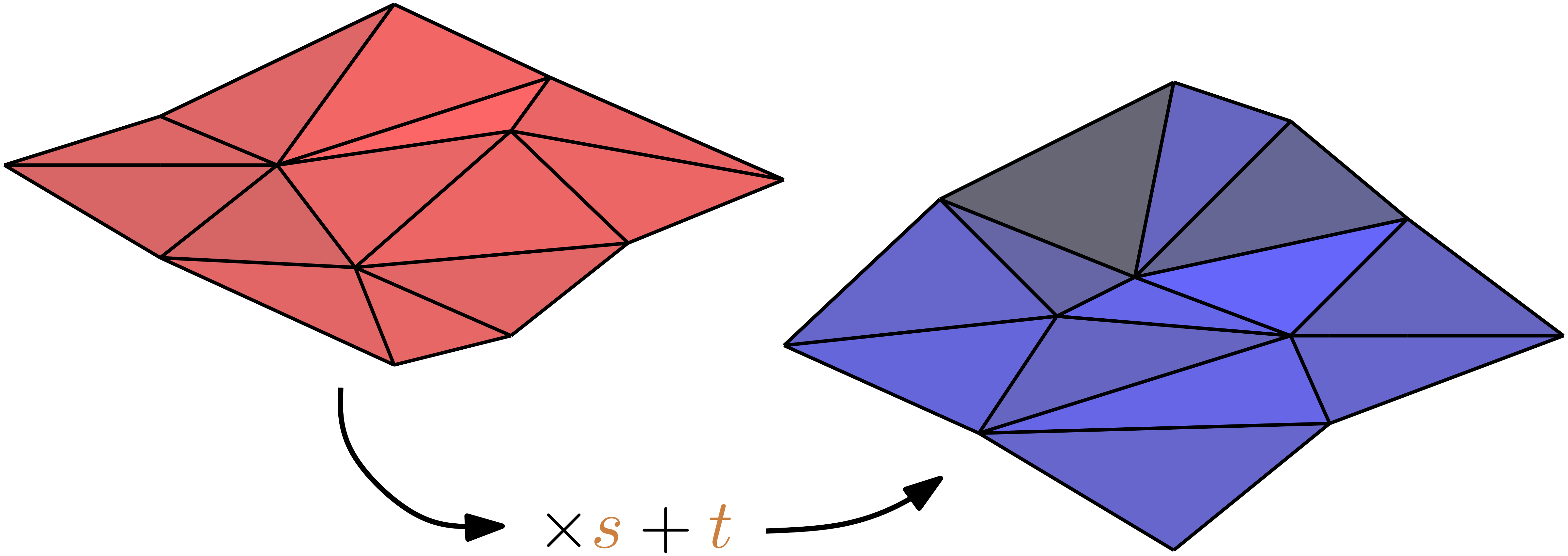


We want to find the translation and scaling of f to align it with g as good as possible.

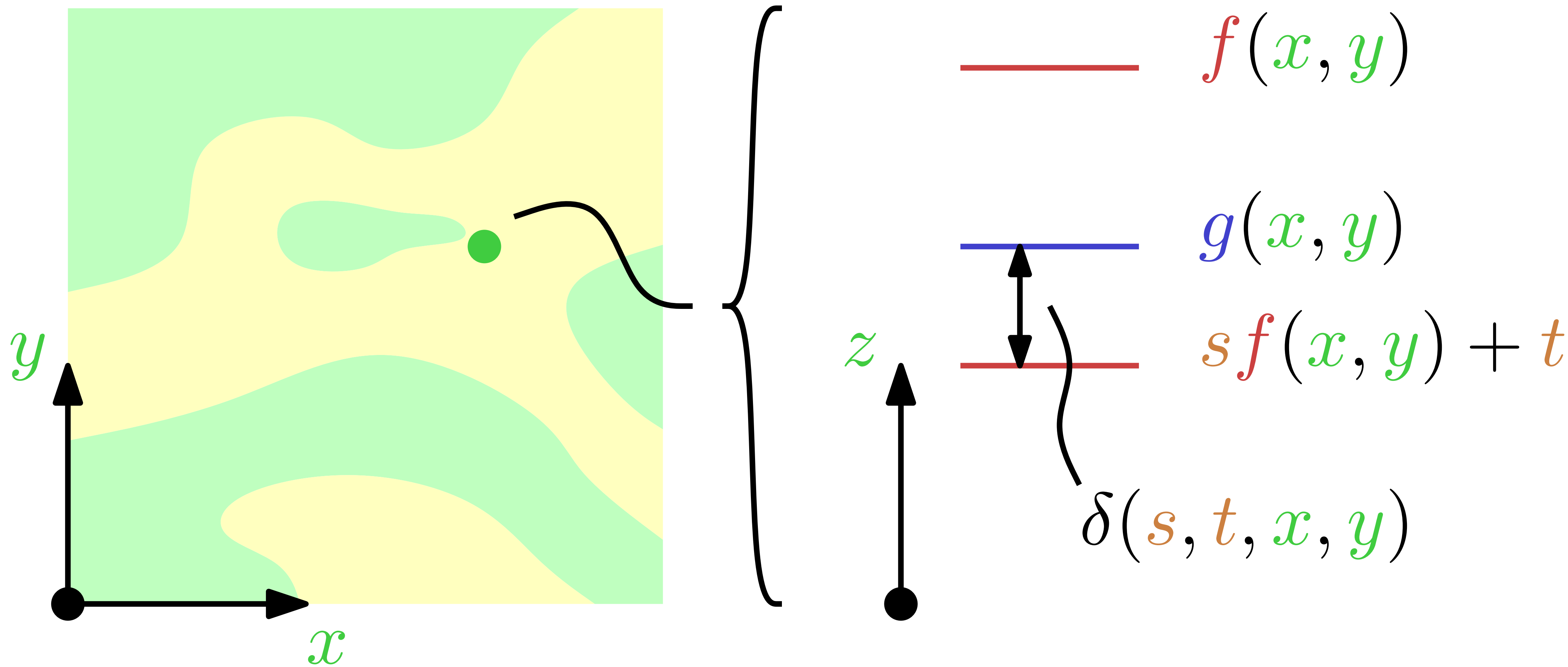


We want to find the translation and scaling of f to align it with g as good as possible.

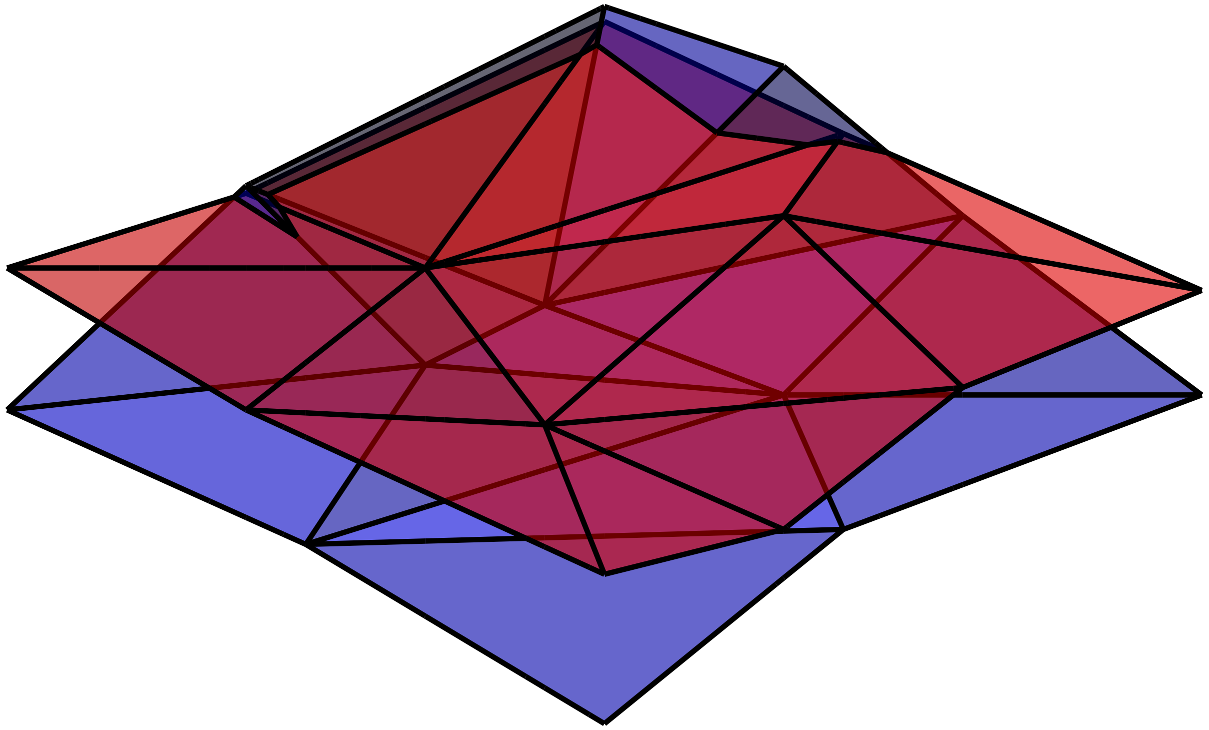
How do we measure the similarity between two TINs?



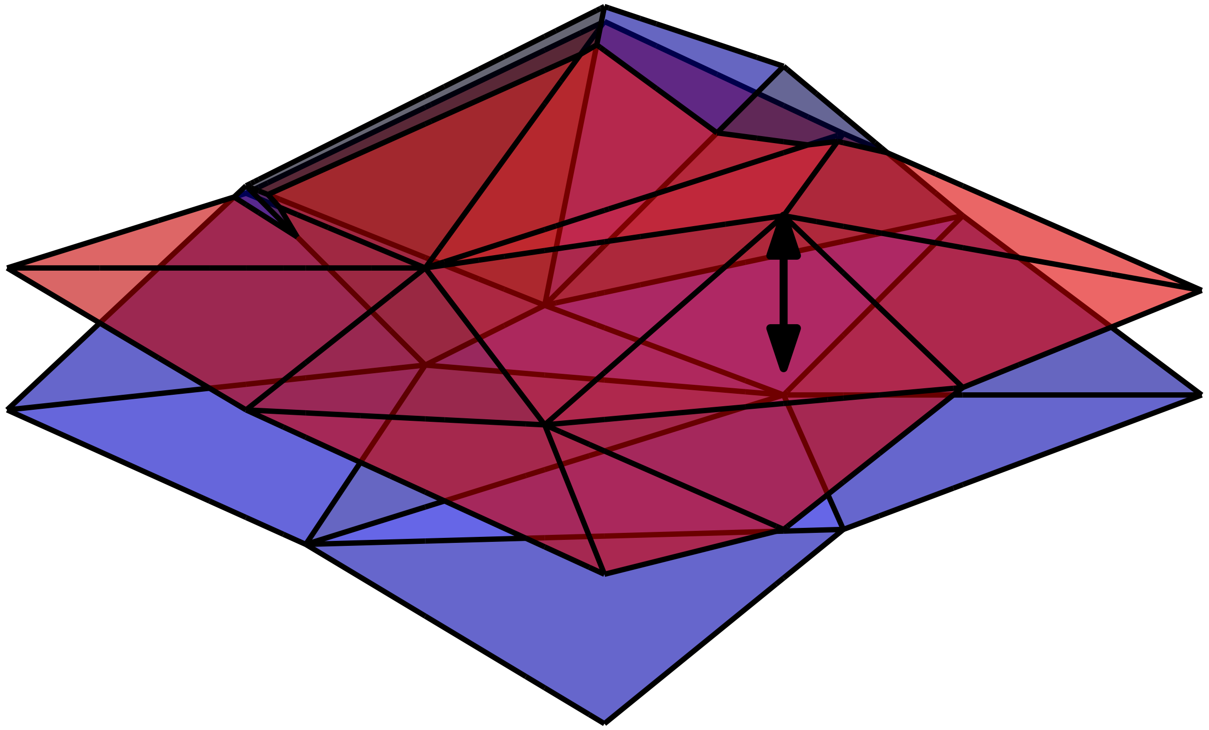
We want to find the two parameters s and t such that $sf + t$ is similar to g.

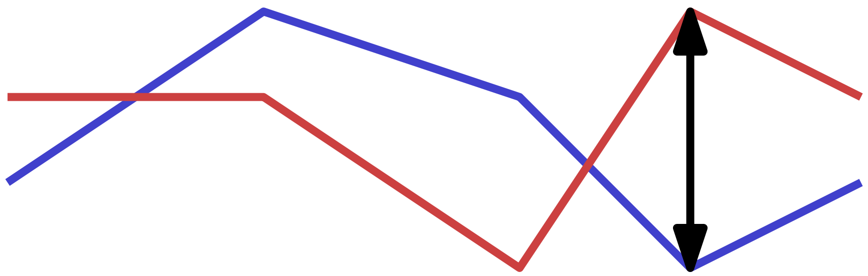


We parameterise the domain by x and y. For each s, t and x, y we can measure the vertical distance between f and g.



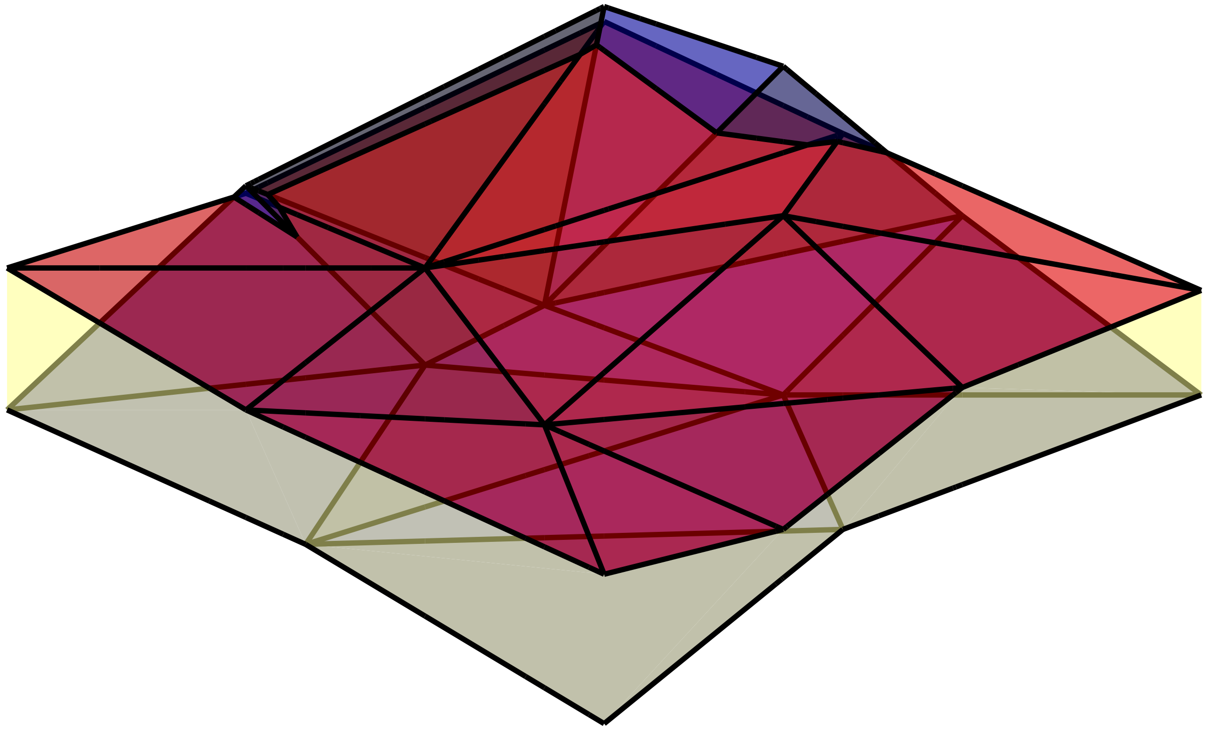
For a given s and t, we can measure either the *worst* or the *average* vertical distance between f and g over all x and y.

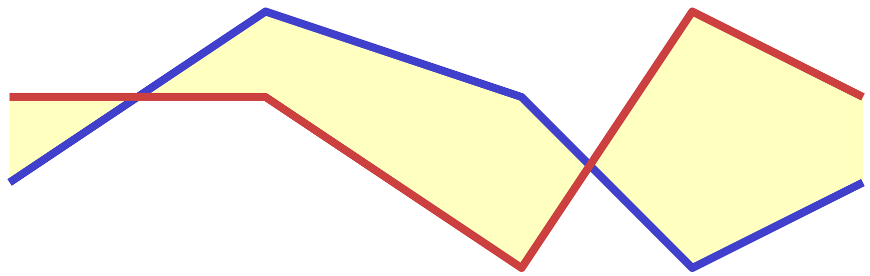




$$\mu_m(s,t) = \max_{x,y} |sf(x,y) + t - g(x,y)|$$

The worst vertical distance is the point on the domain where the two TINs are furtherst away from each other.





$$\mu_a(s,t) = \int_{x,y} |sf(x,y) + t - g(x,y)|\,dx\,dy$$

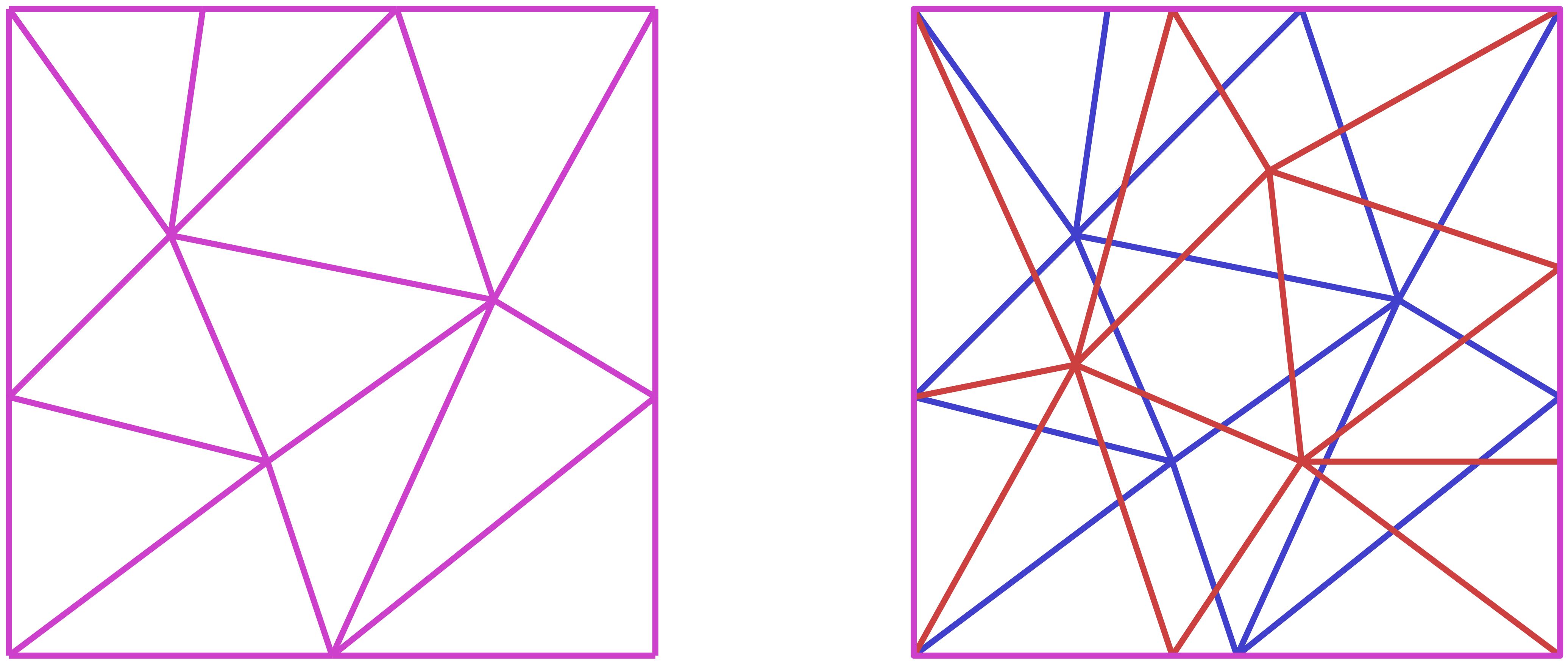
Optimising the average vertical distance is the same as optimising the integral, or the *volume* between the two TINs.

$$\min_{s,t} \mu_m(s, t) = ?$$

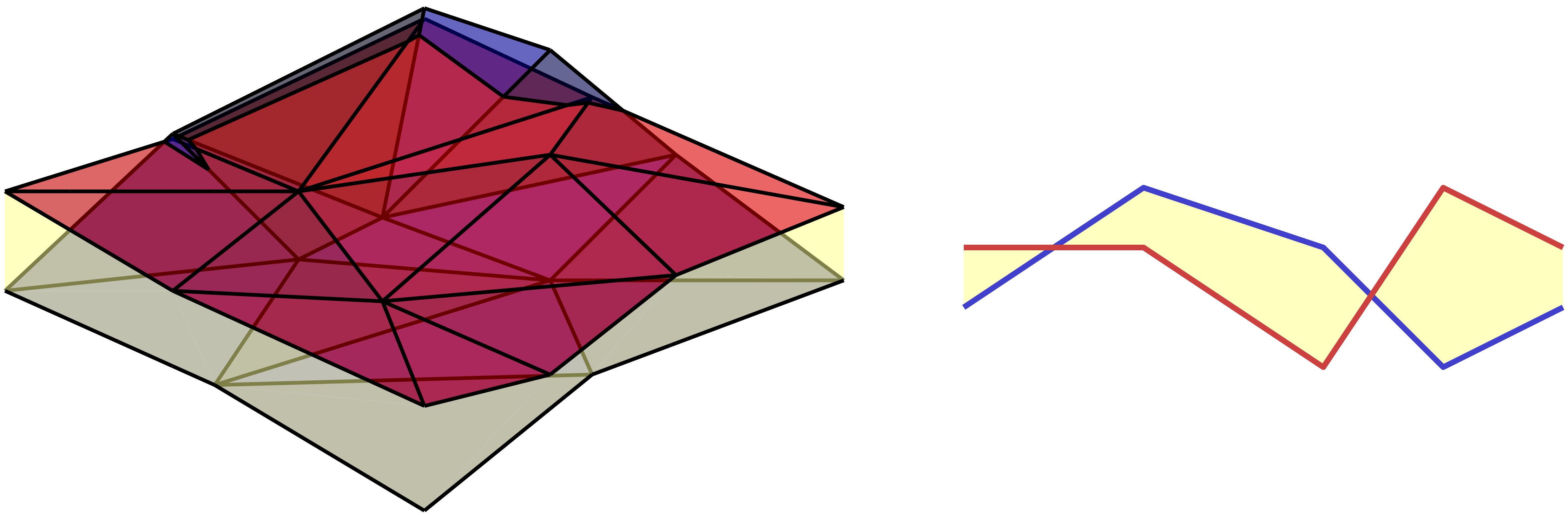
$$\min_{s,t} \mu_a(s, t) = ?$$

We want to minimise μ_m or μ_a over all s and t.

What results did we get?

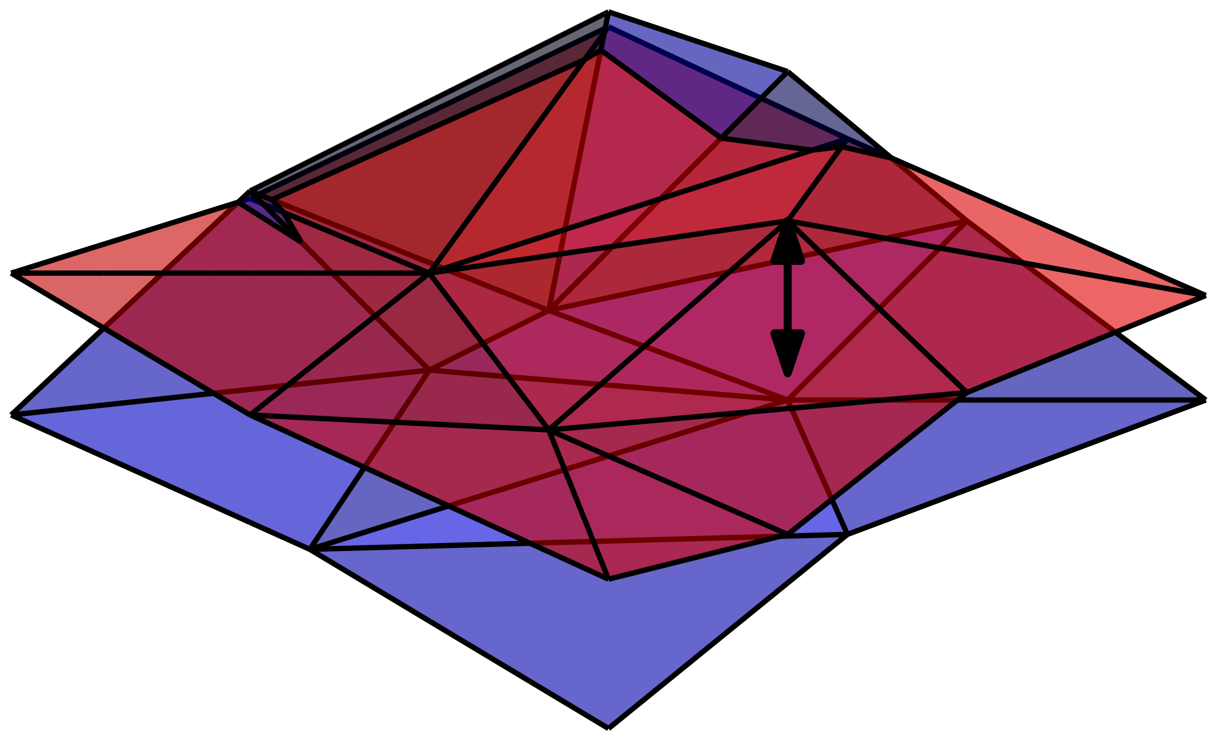


We can minimise μ_m over all s and t in $O(n)$ time if the terrains are aligned, and in $O(n^{4/3}\text{polylog } n)$ otherwise.

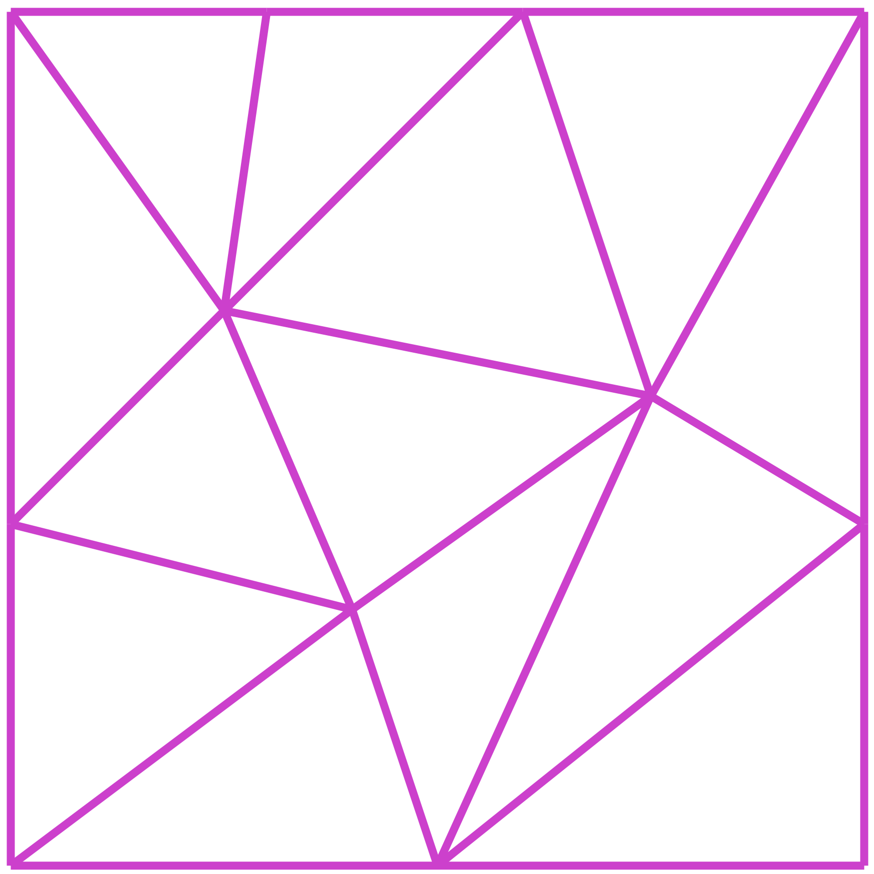


We cannot minimise μ_a. But for 1.5-dimensional terrains, we can compute a $(1+\varepsilon)$-approximation in $O(n/\sqrt{\varepsilon})$ time.

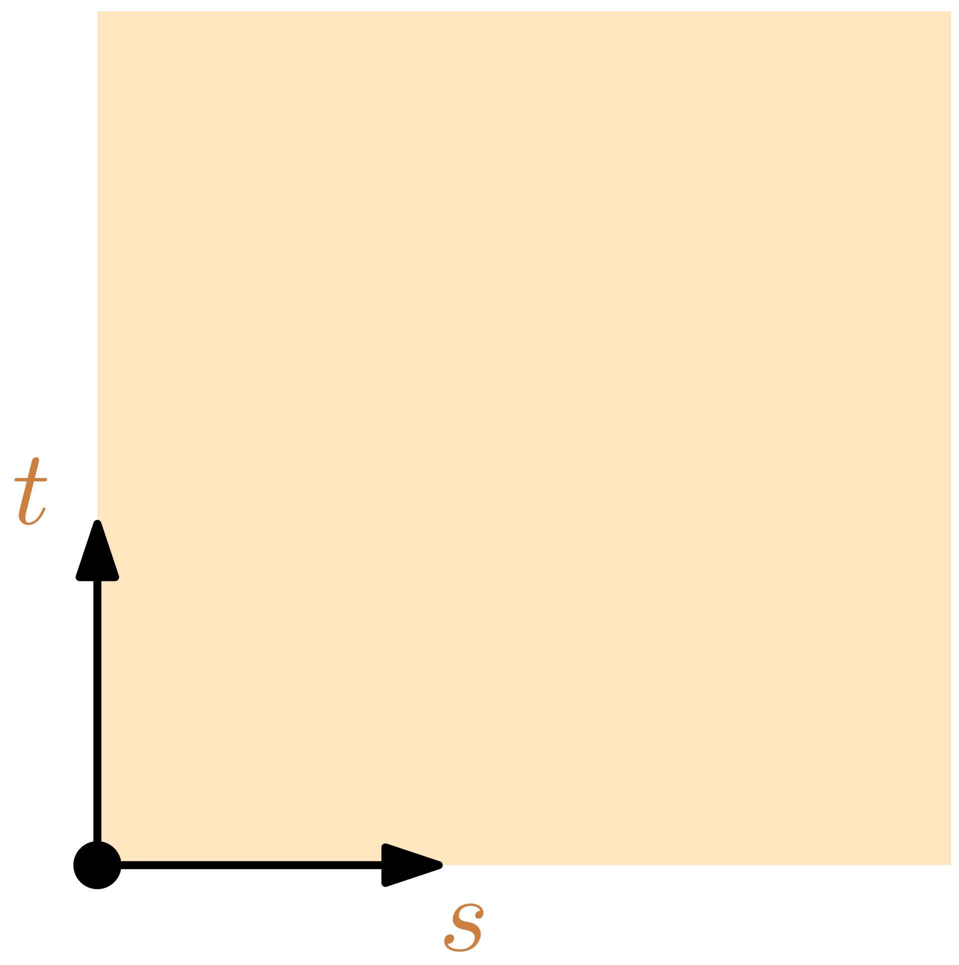
How do we minimise $\mu_m(s, t)$ when the terrains are aligned?



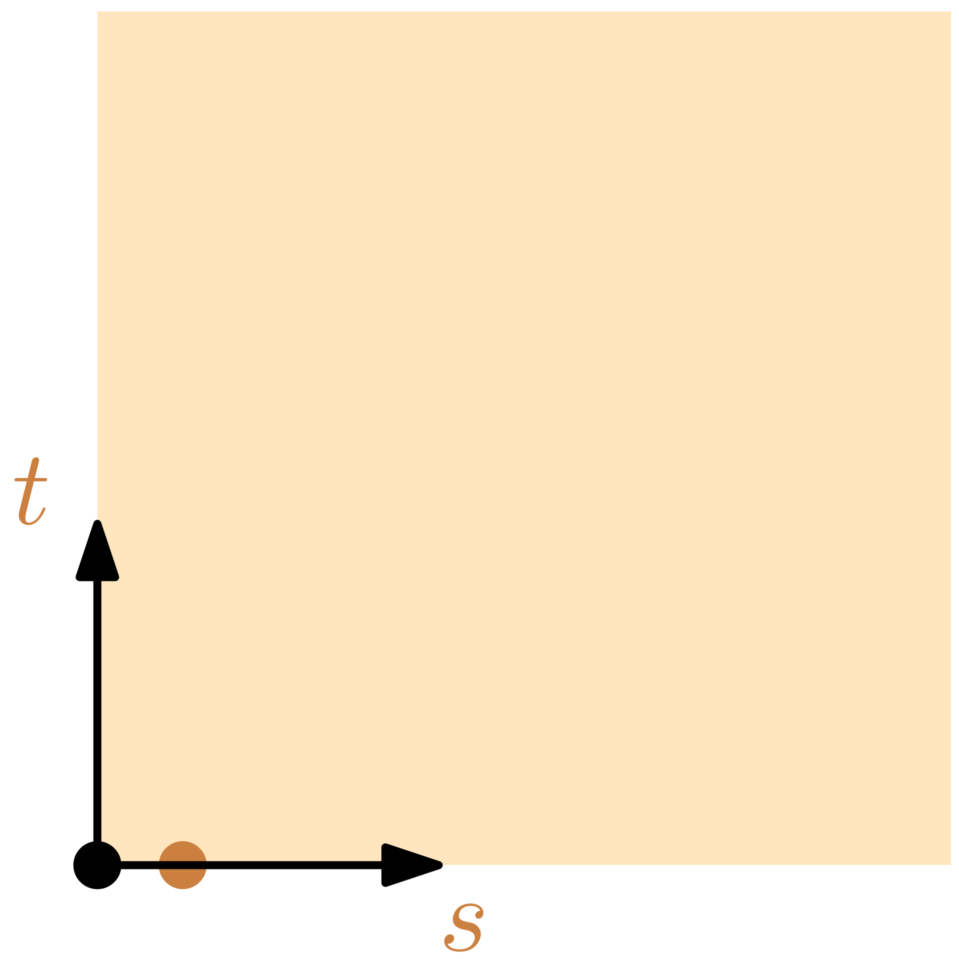
For fixed s and t, the worst vertical distance always occurs at a vertex of the overlay of the TINs.

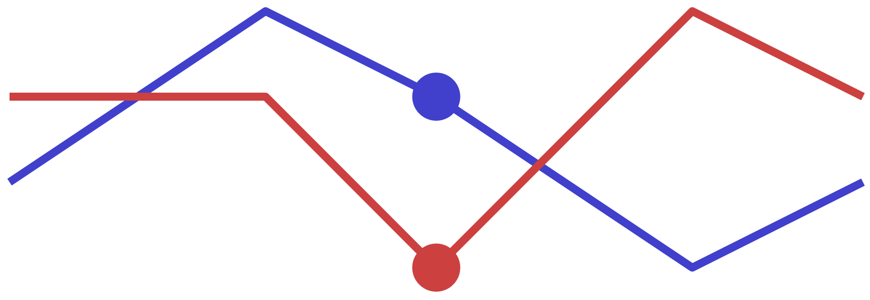


When the terrains are aligned, there are n such vertices.

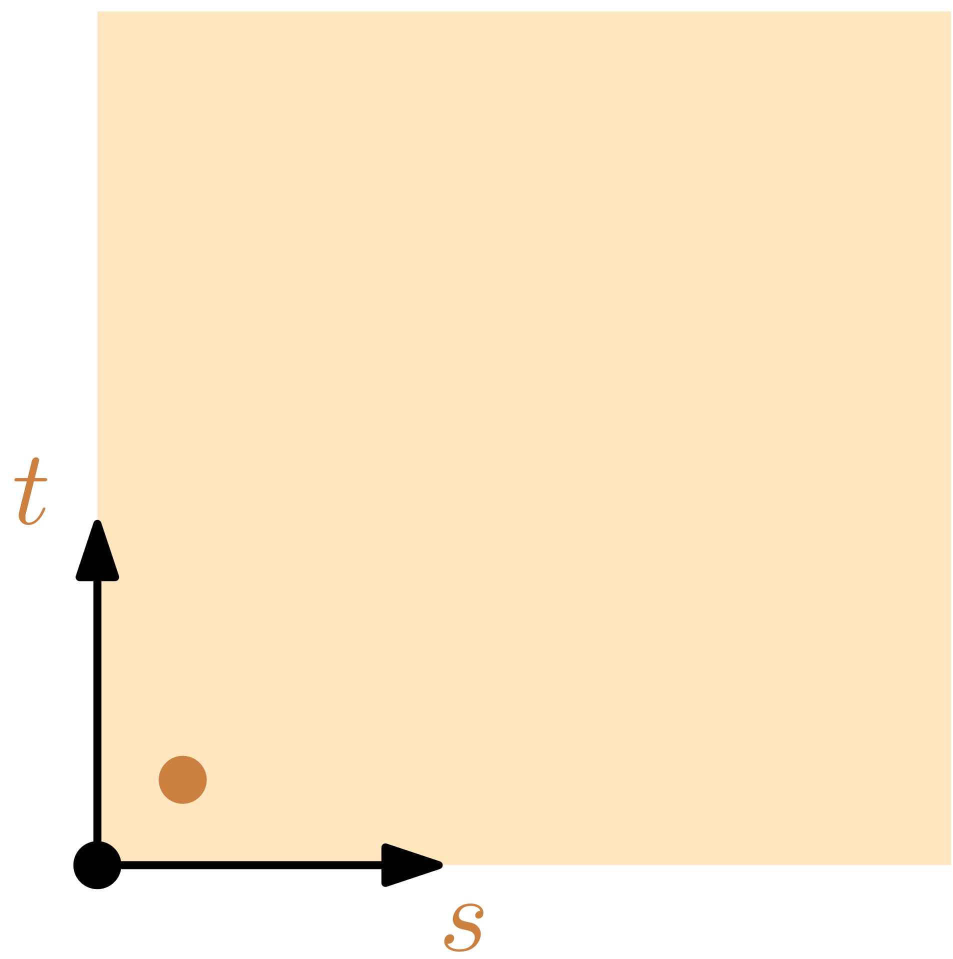


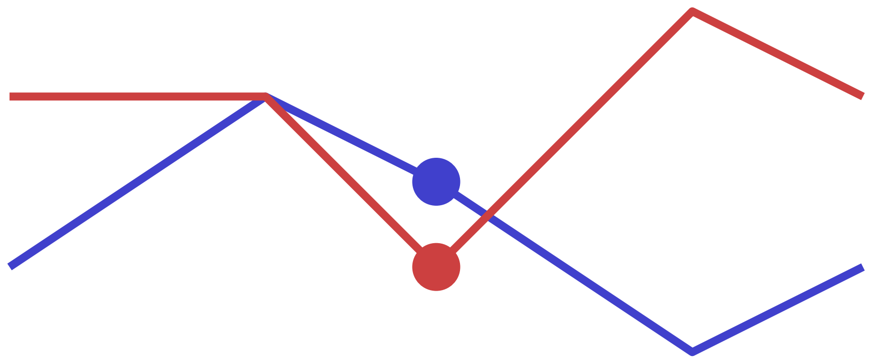
Consider the parameter (s, t)-space. At each point, $\mu_m(s, t)$ is defined as the maximum over all vertices (x, y).



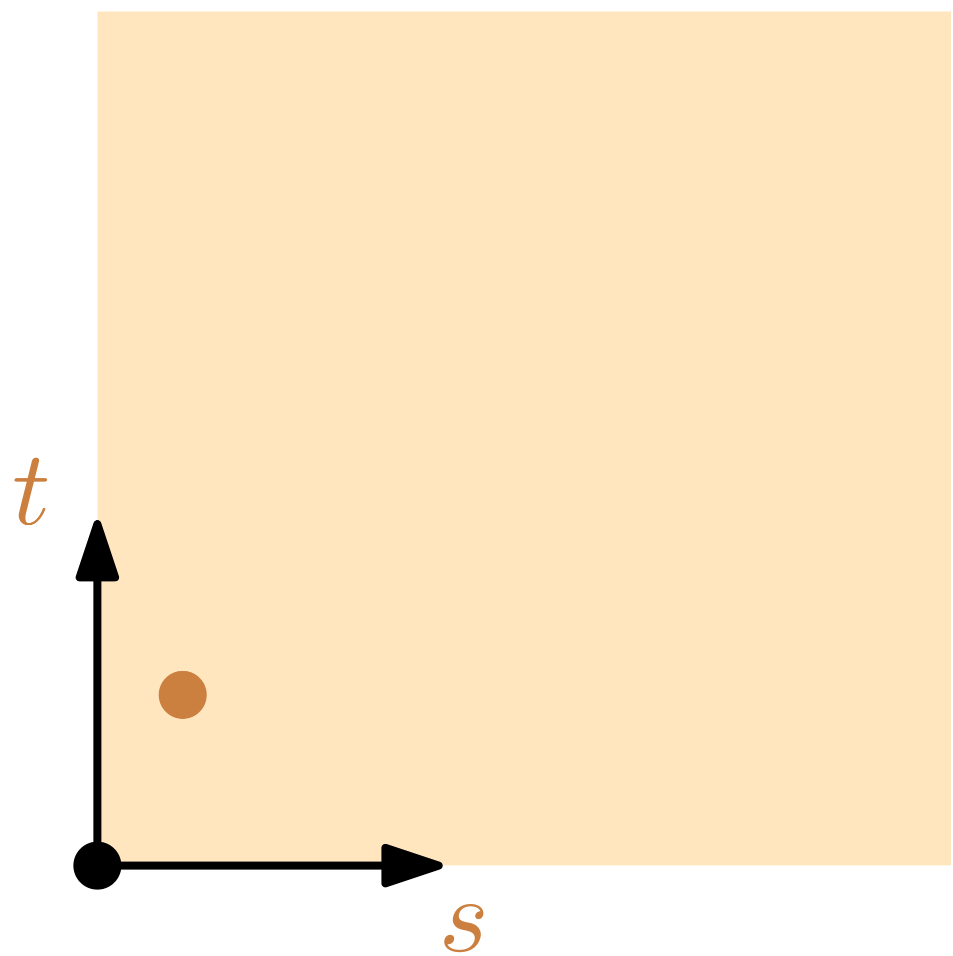


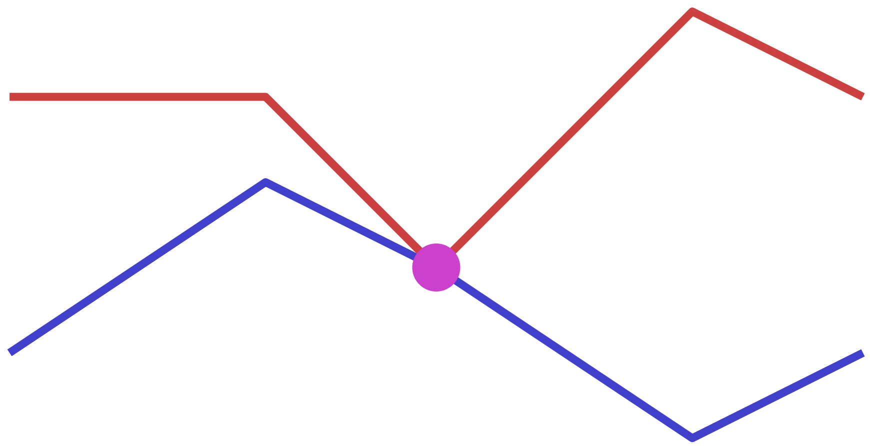
A blue and red vertex with the same projection have vertical distance 0 when they coincide.



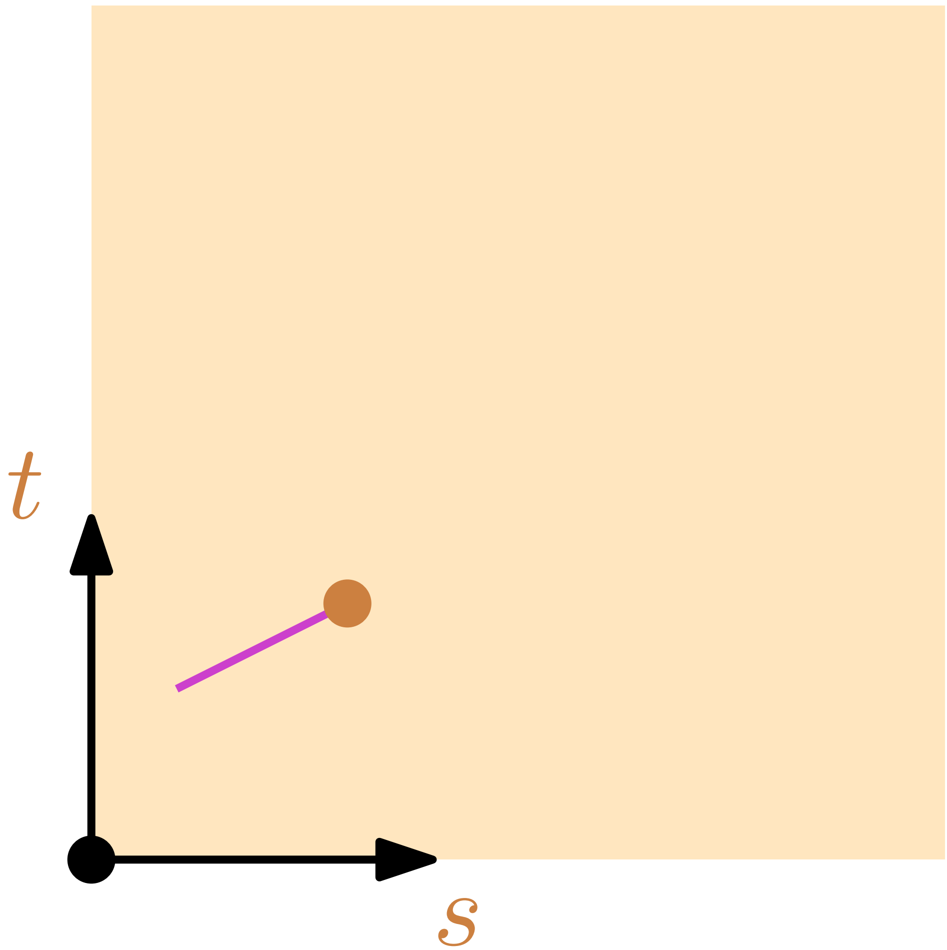


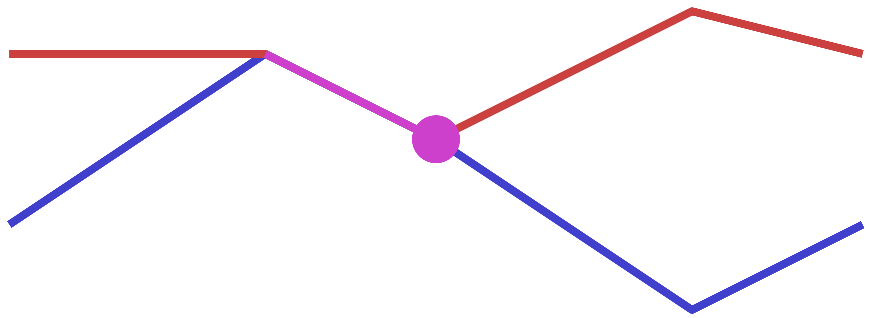
A blue and red vertex with the same projection have vertical distance 0 when they coincide.



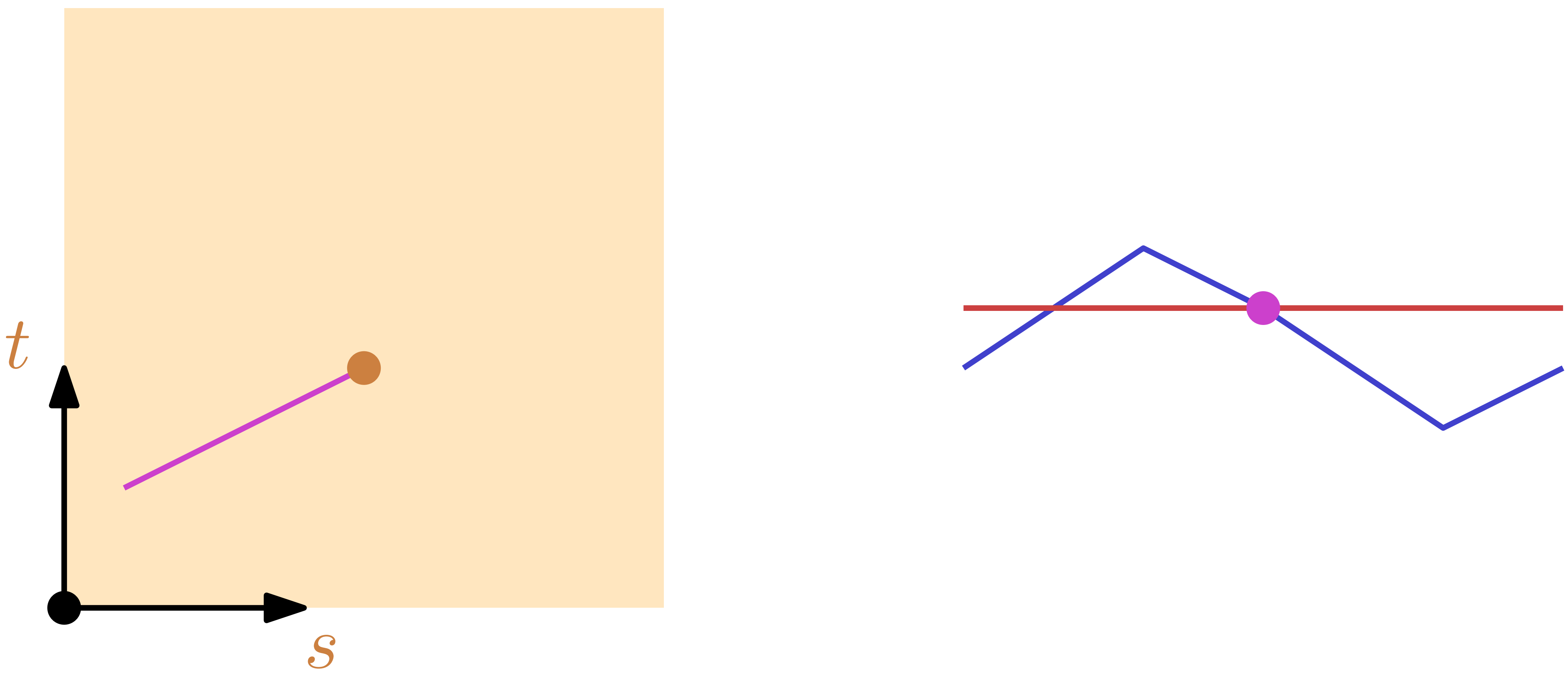


A blue and red vertex with the same projection have vertical distance 0 when they coincide.

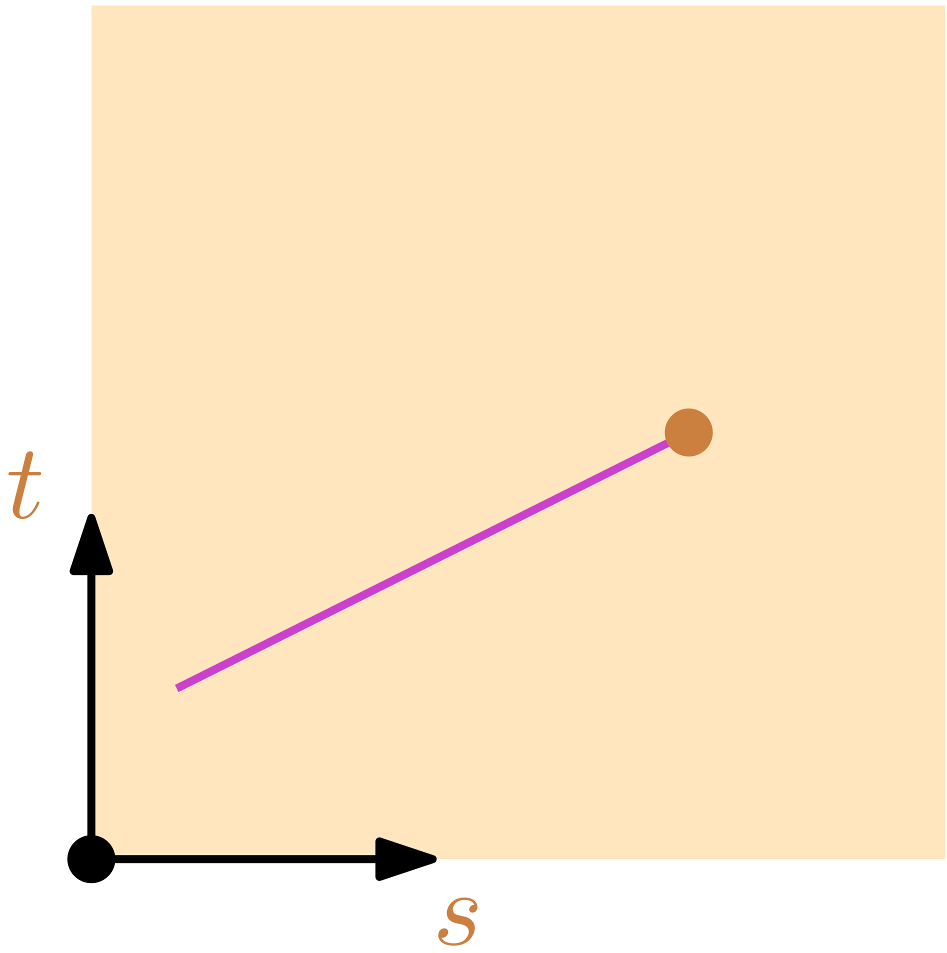


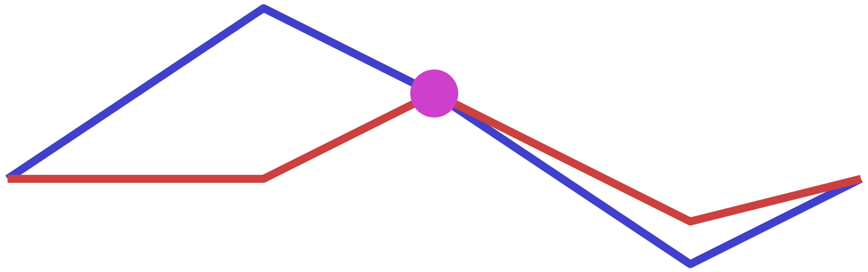


This defines a line in (s, t)-space.

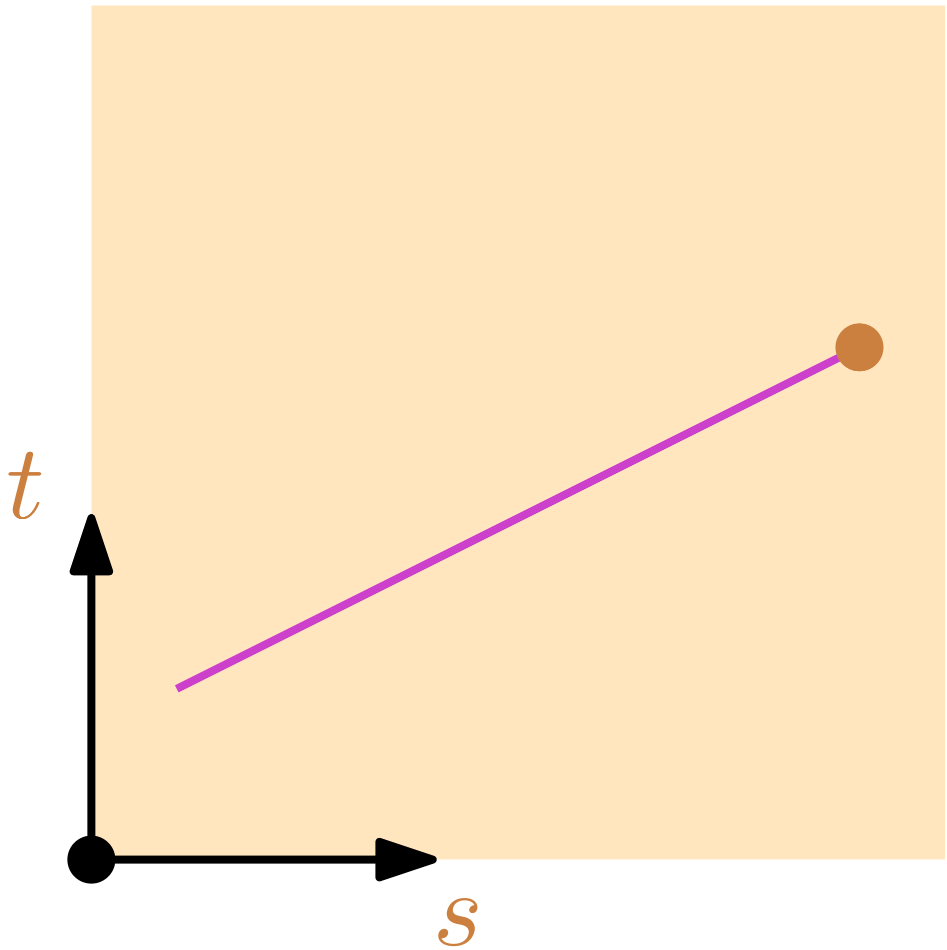


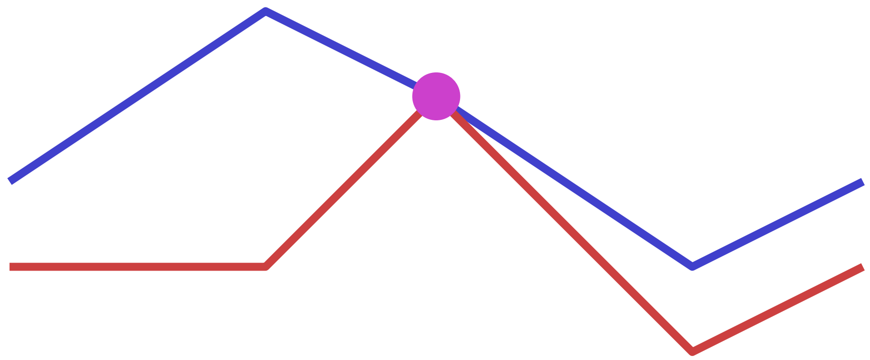
This defines a line in (s, t)-space.



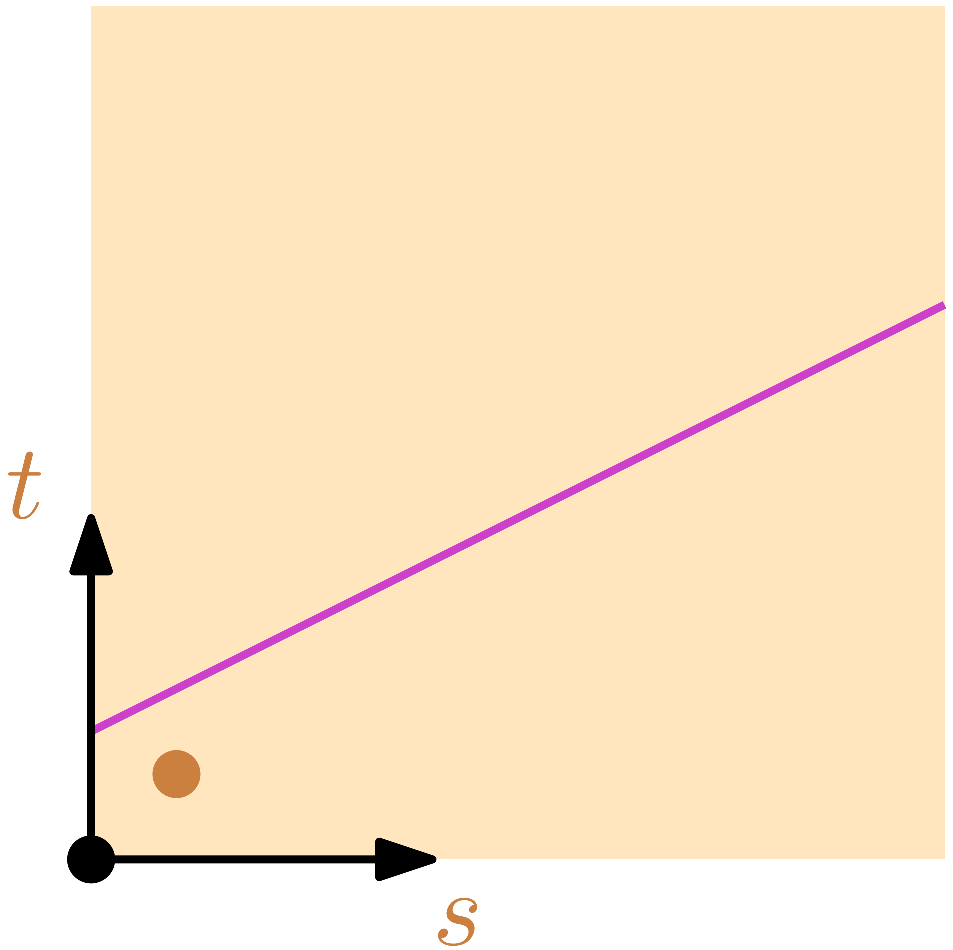


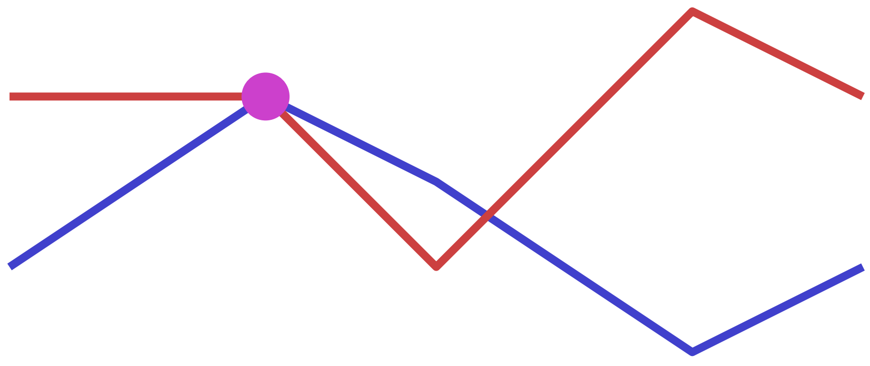
This defines a line in (s, t)-space.



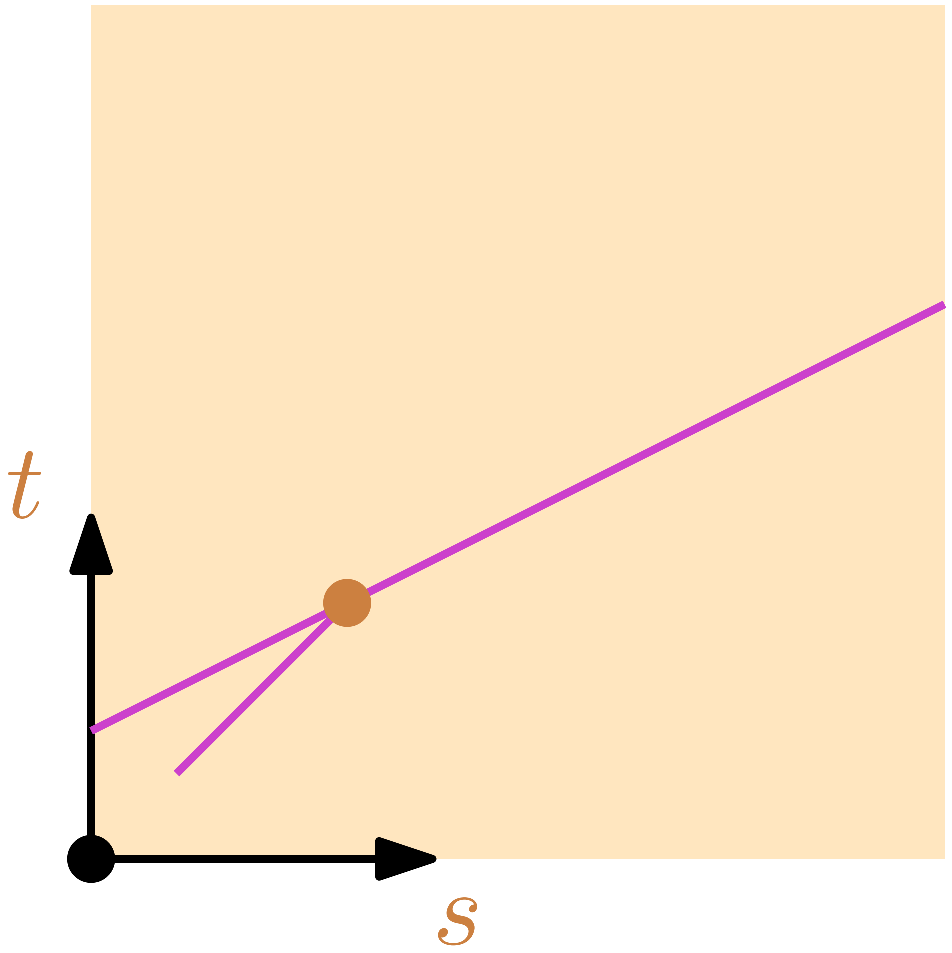


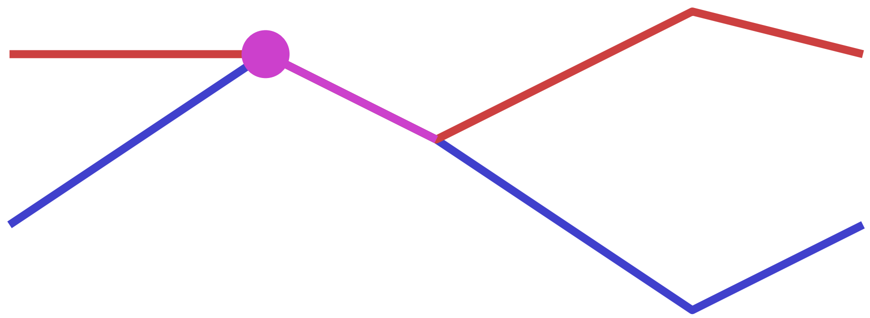
This defines a line in (s, t)-space.



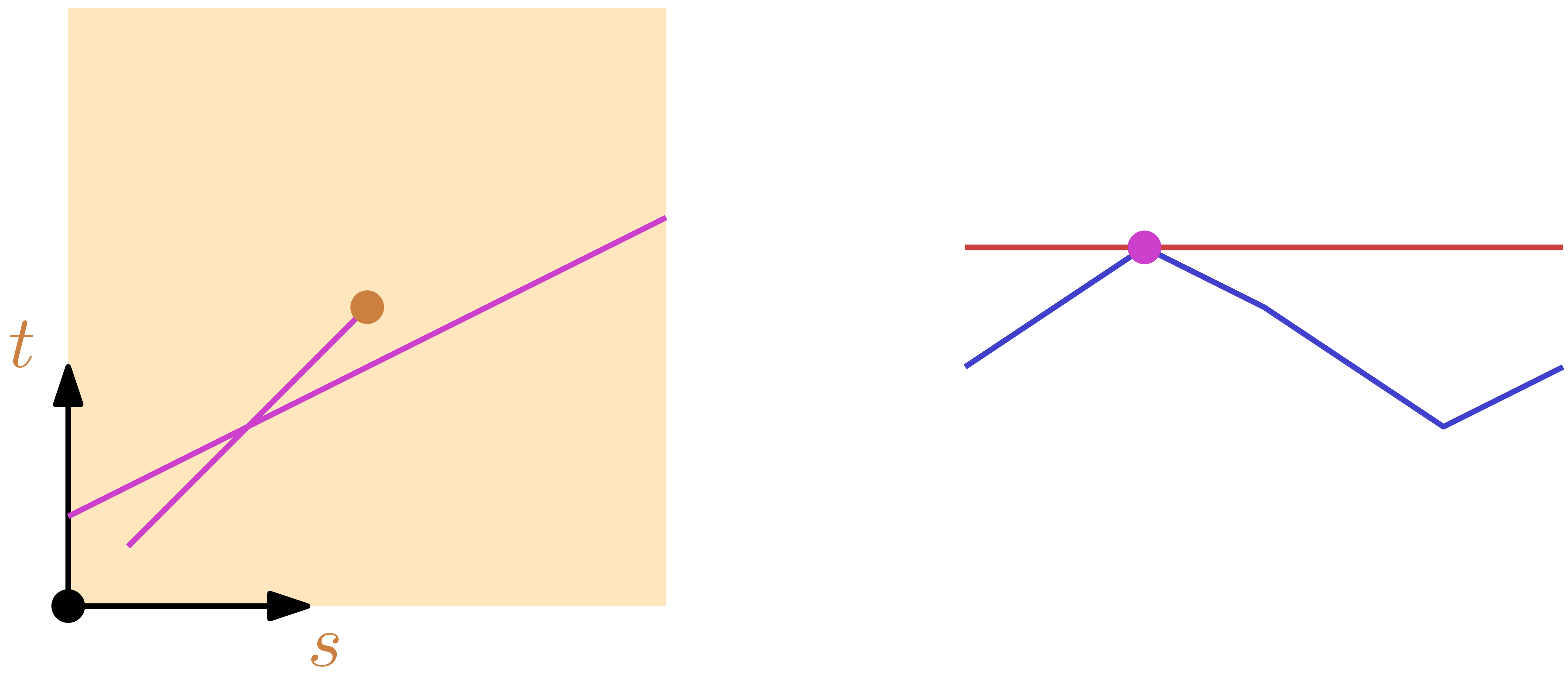


This defines a line in (s, t)-space.

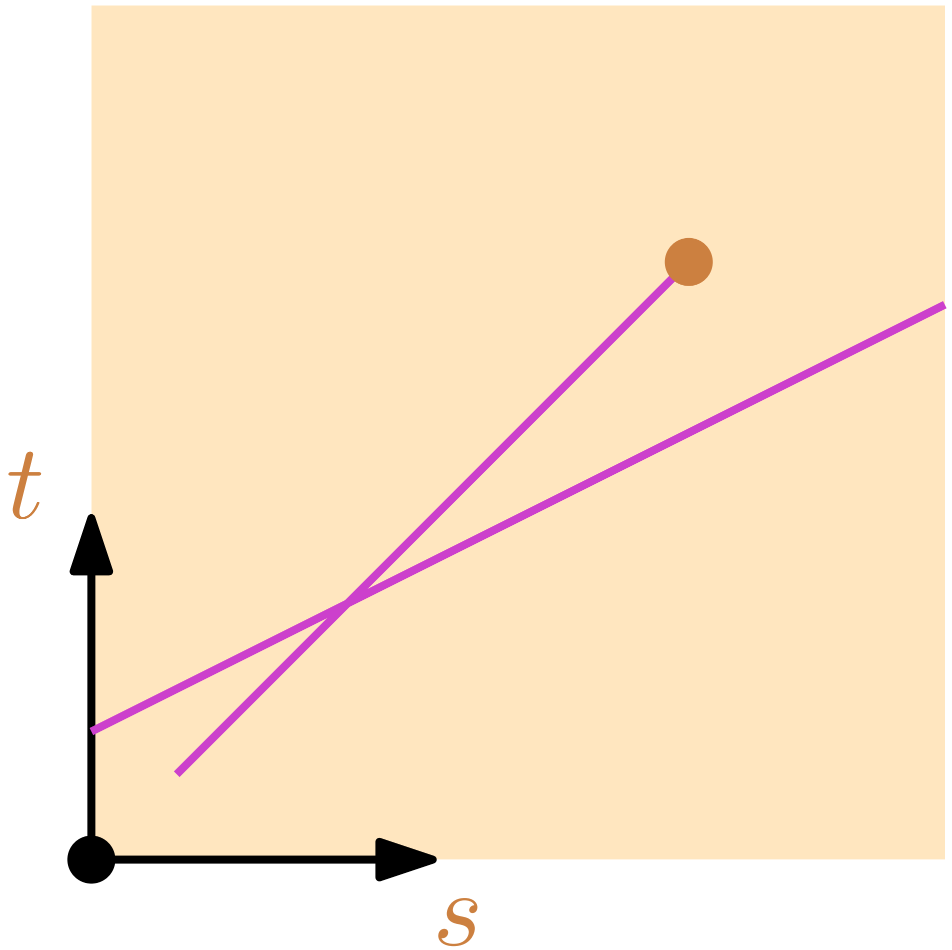


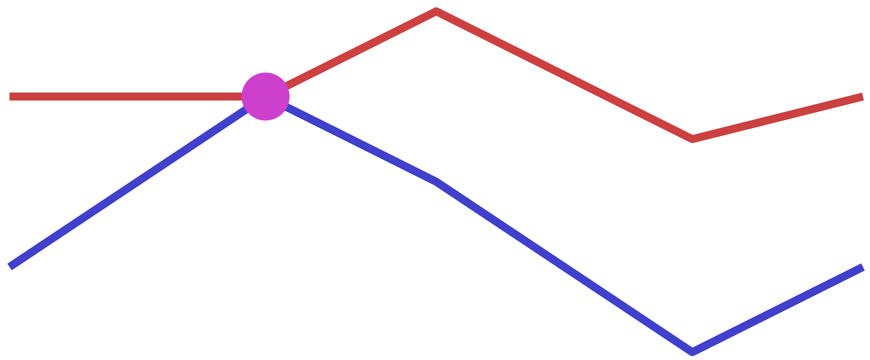


This defines a line in (s, t)-space.

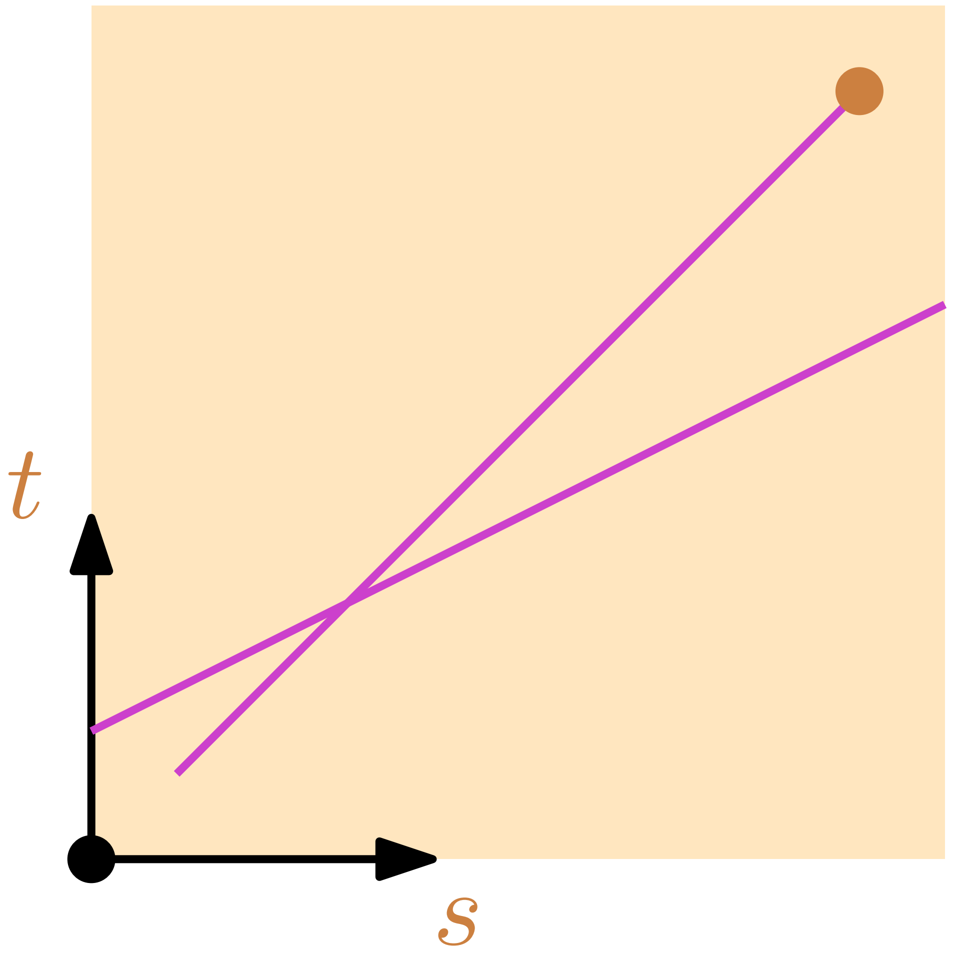


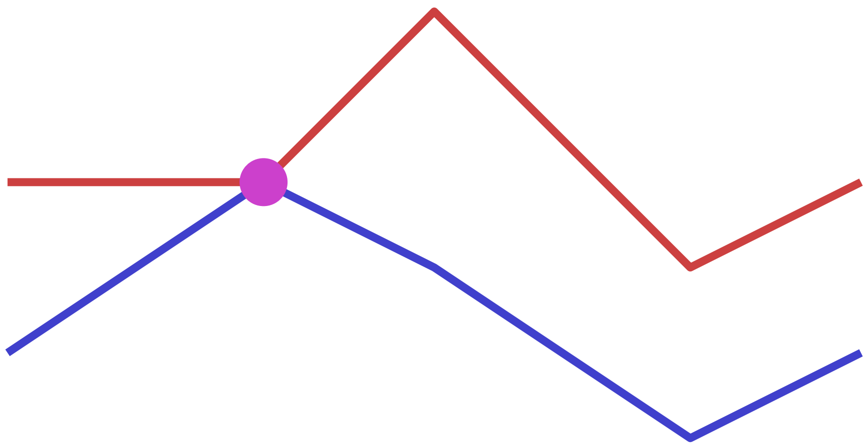
This defines a line in (s, t)-space.



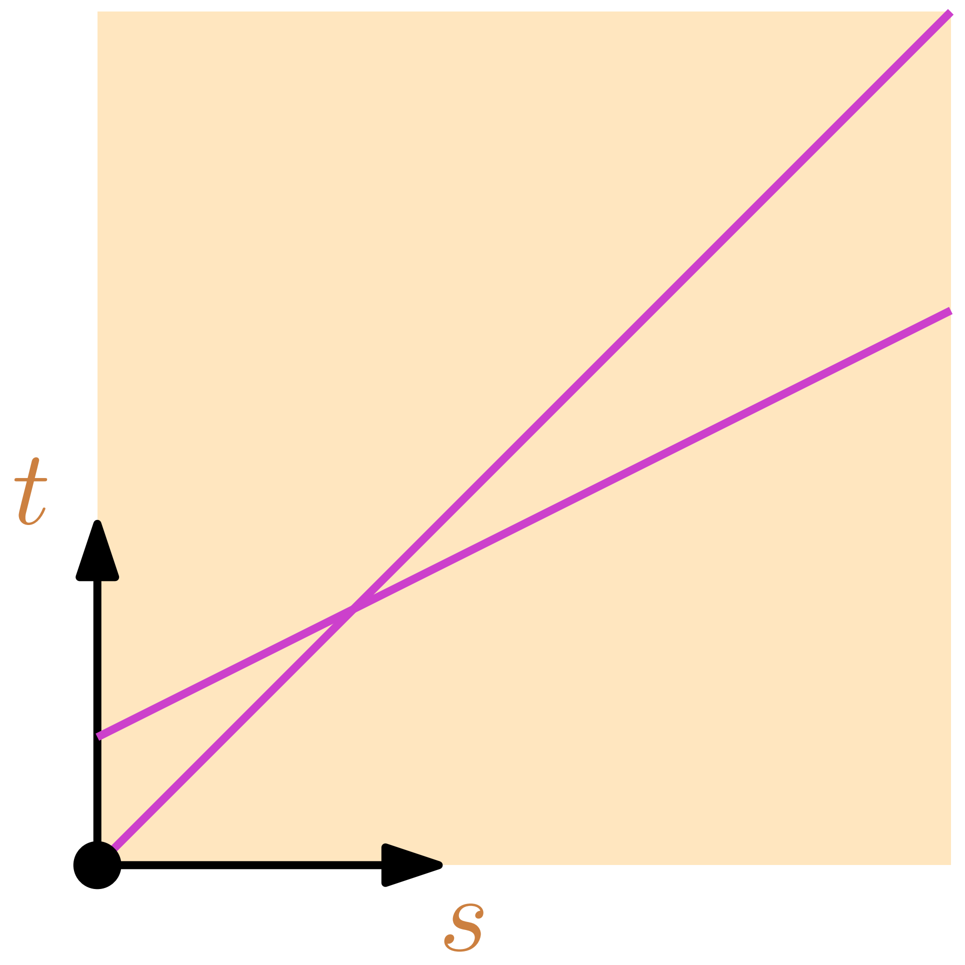


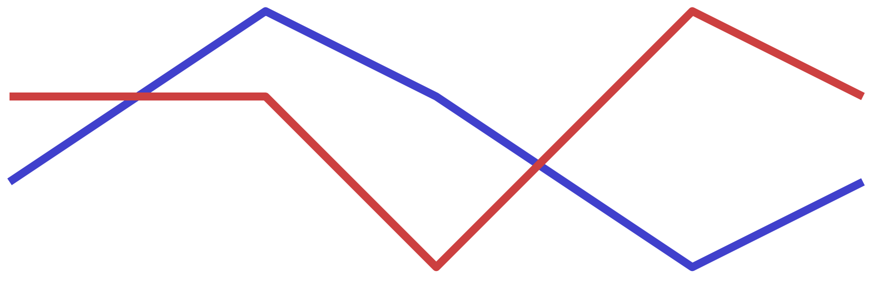
This defines a line in (s, t)-space.



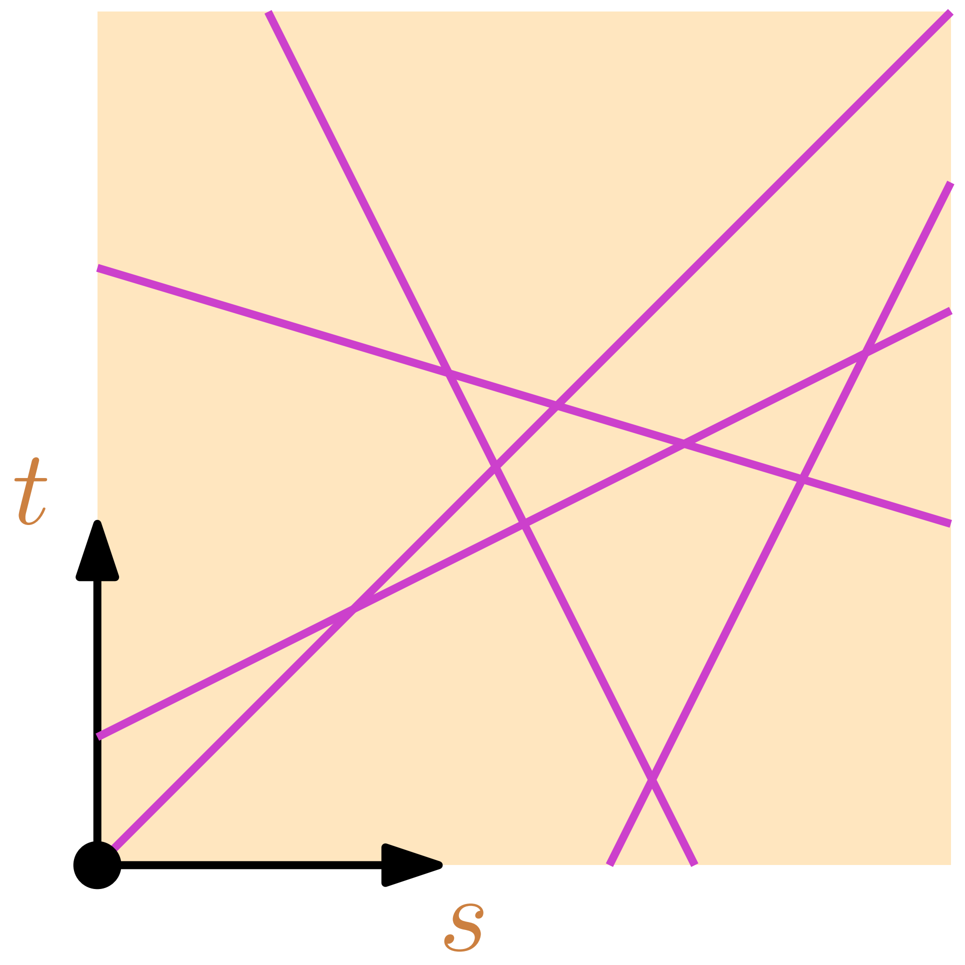


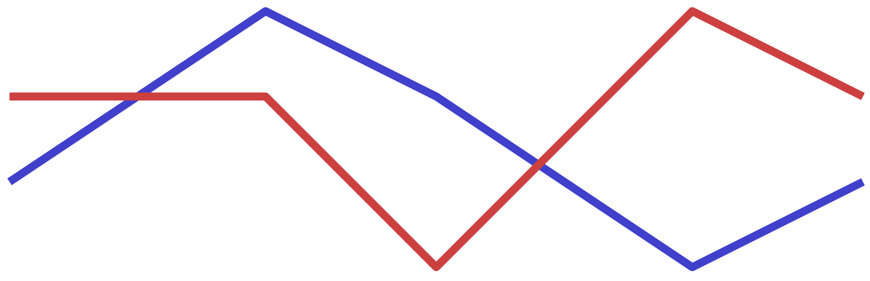
This defines a line in (s, t)-space.



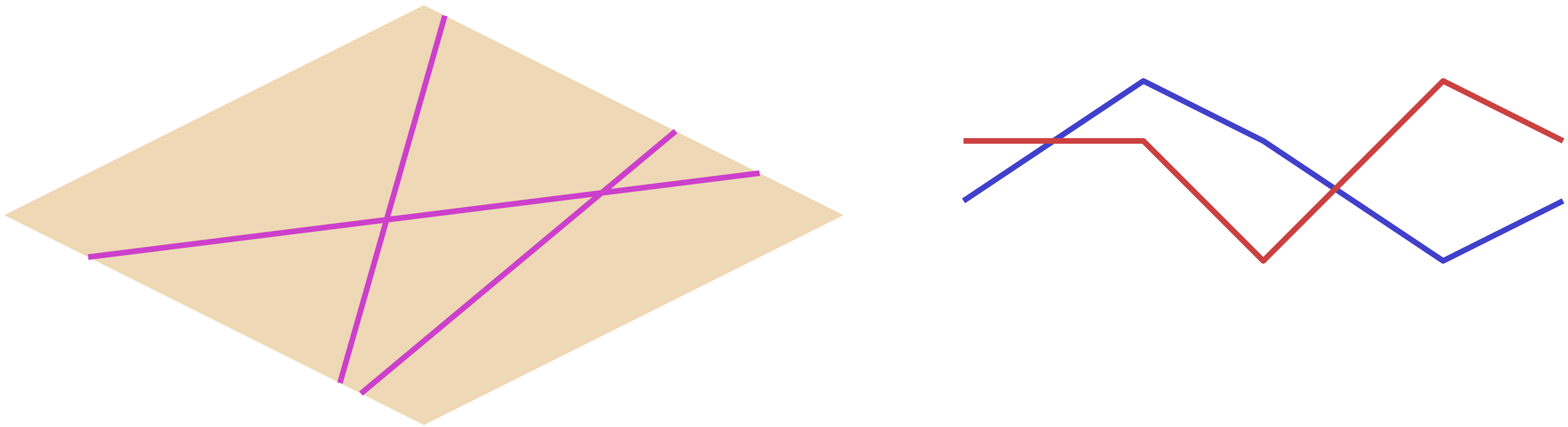


The n vertices of f and g together form an arrangement of n lines.

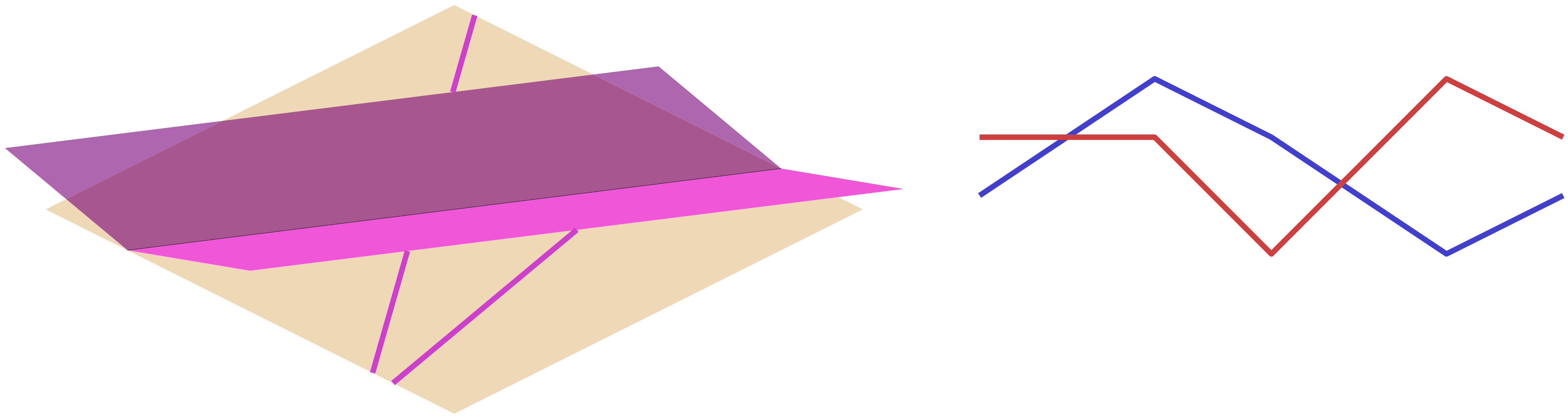




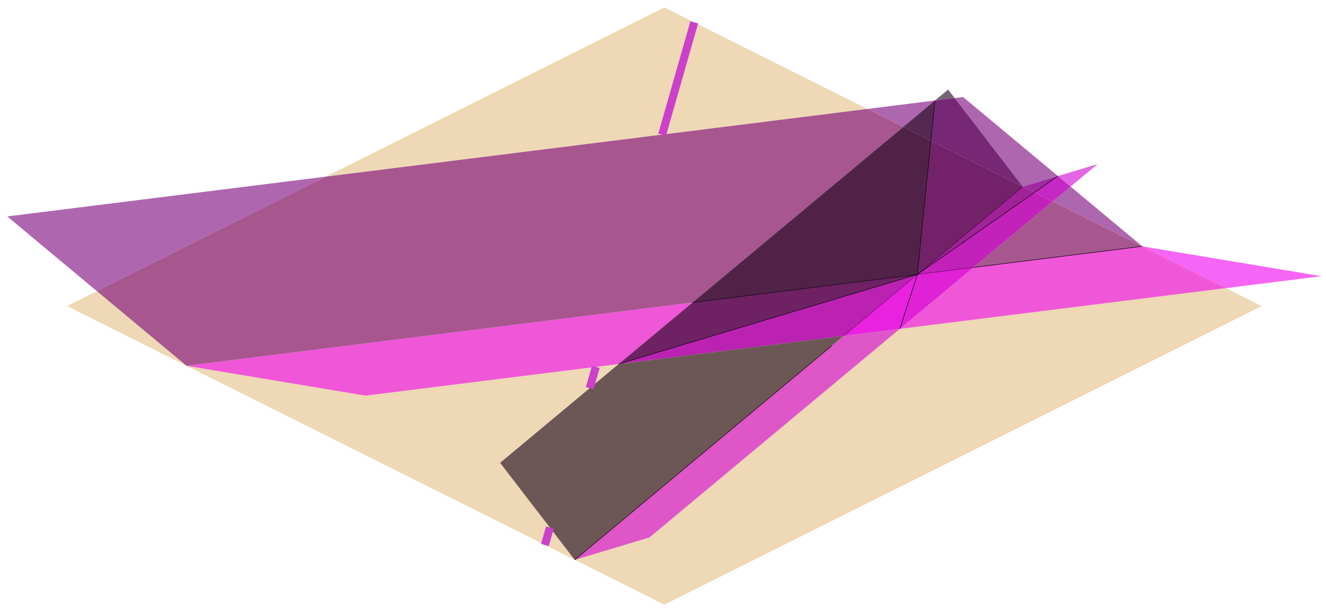
The n vertices of f and g together form an arrangement of n lines.

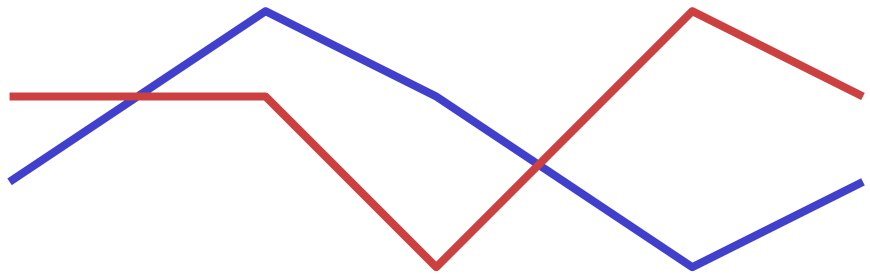


In the (s, t, μ)-space, these lines lie at the intersection of a pair of half-planes.

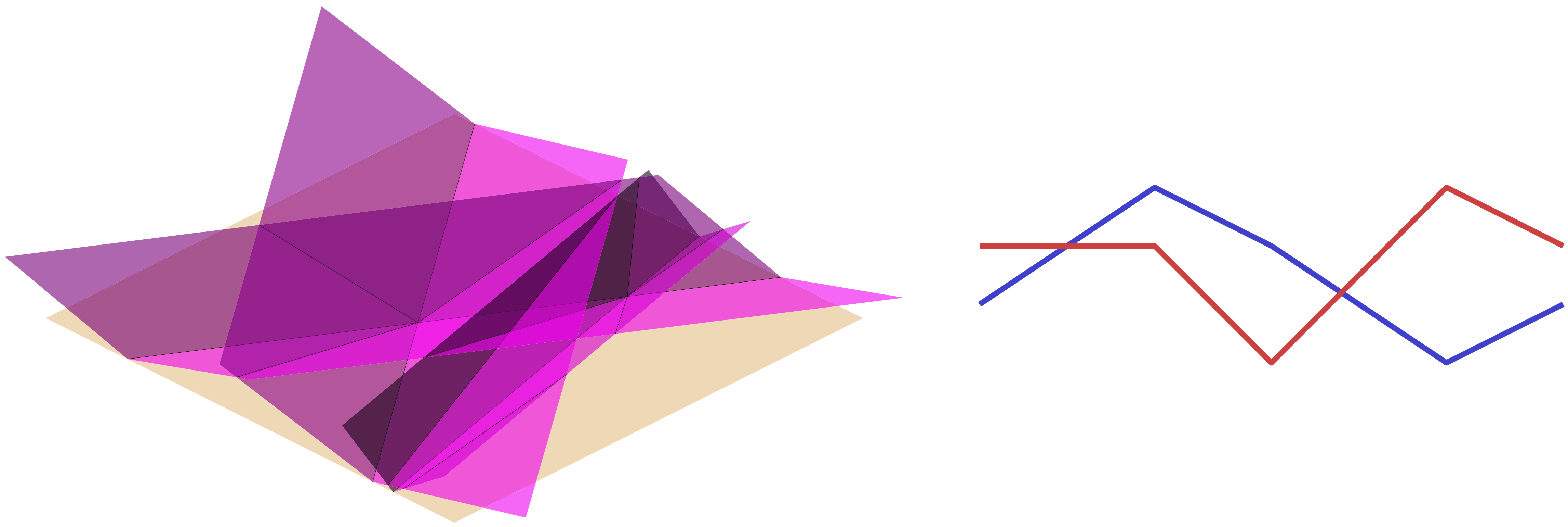


In the (s, t, μ)-space, these lines lie at the intersection of a pair of half-planes.

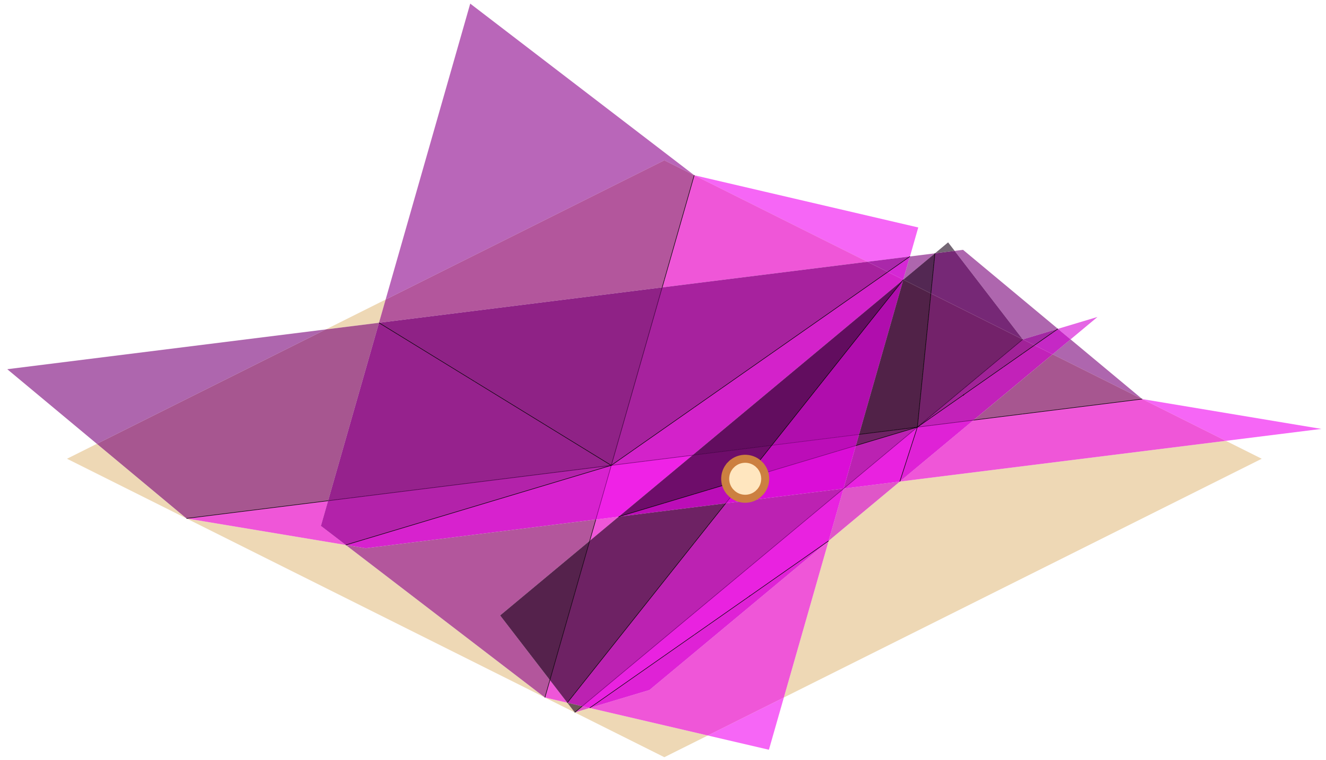


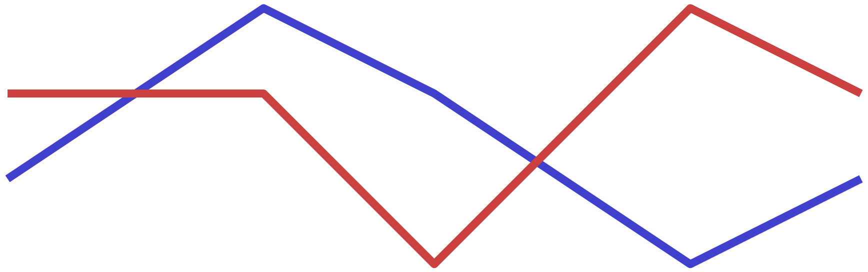


In the (s, t, μ)-space, these lines lie at the intersection of a pair of half-planes.

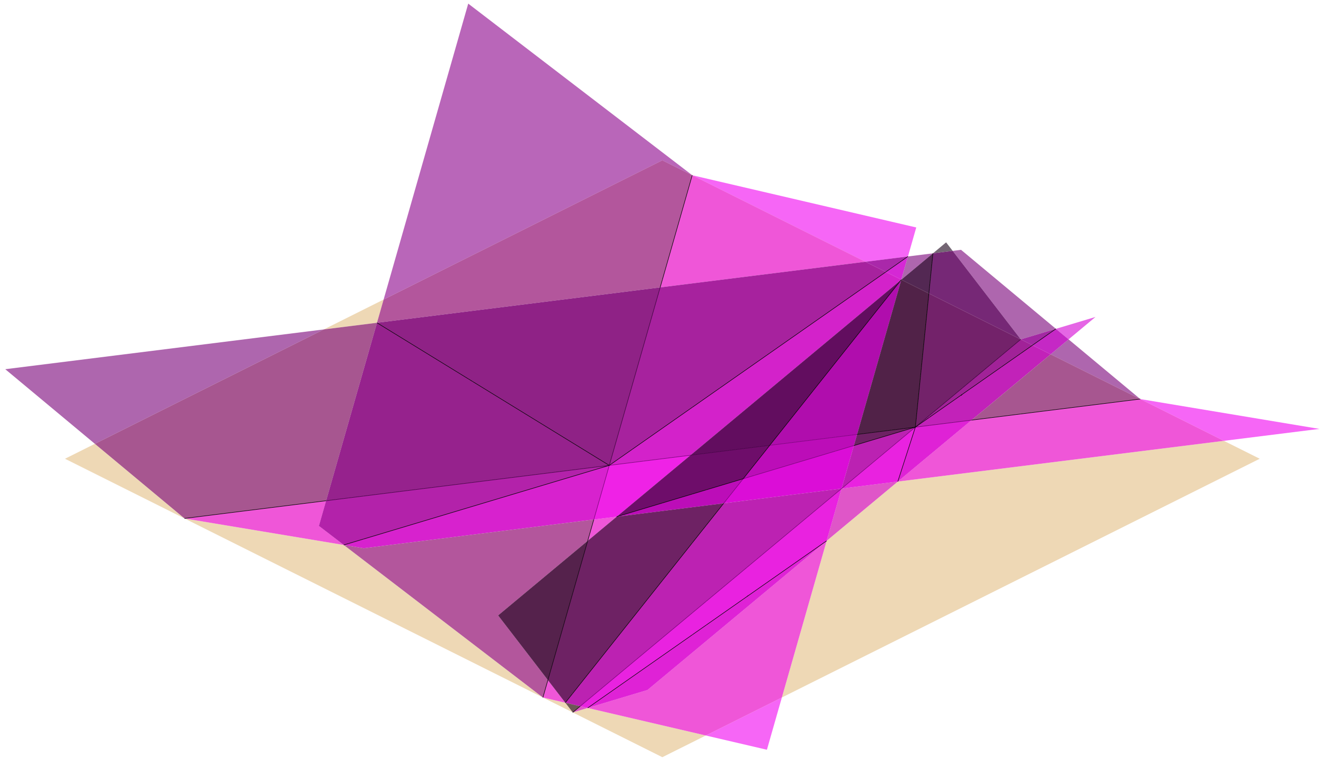
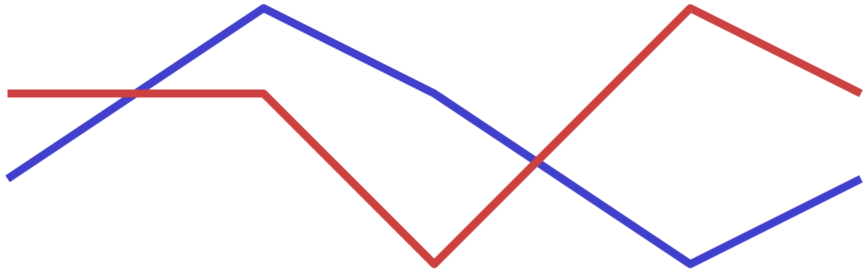


In the (s, t, μ)-space, these lines lie at the intersection of a pair of half-planes.



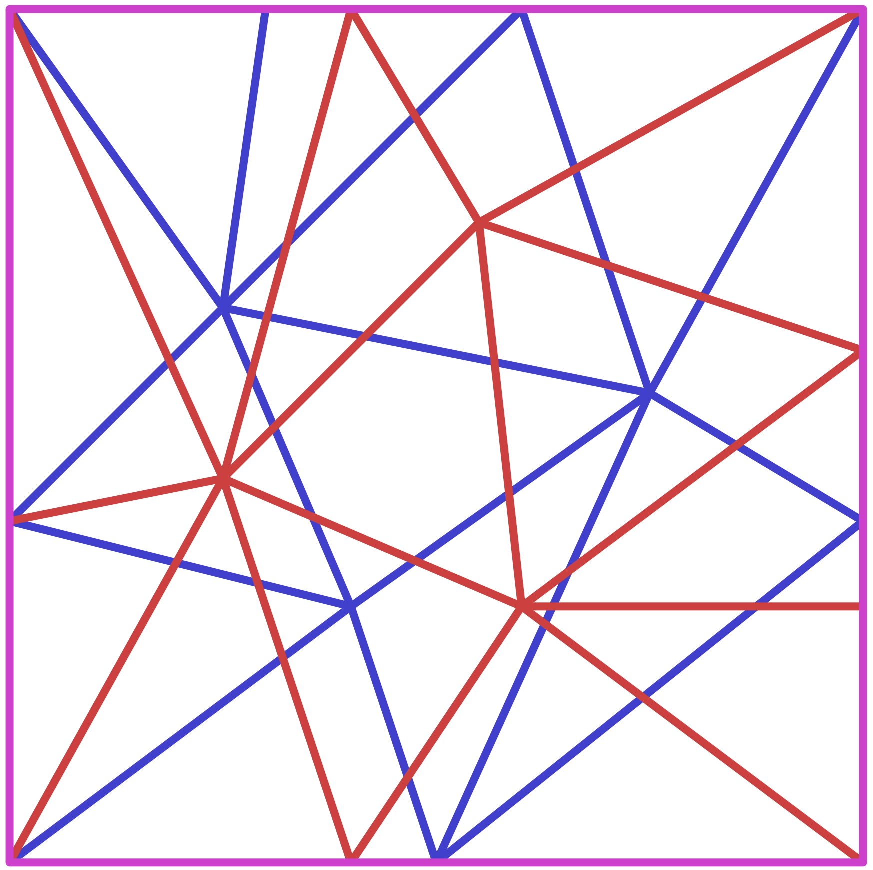


The optimal values of s and t are determined by the lowest point above all half-planes.

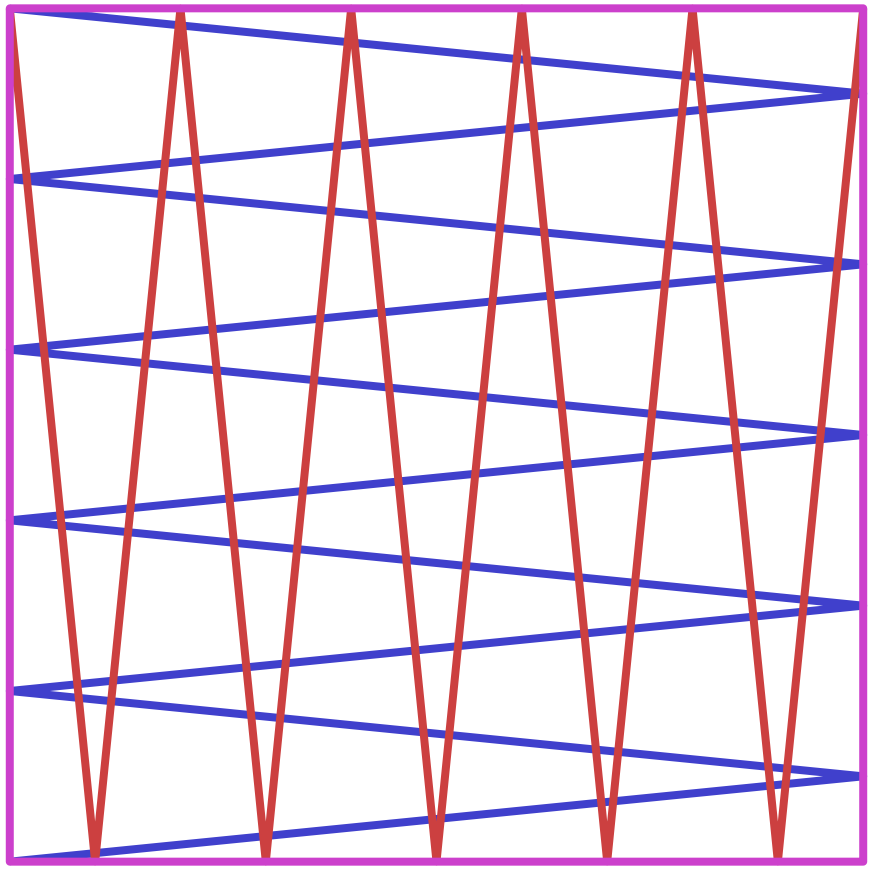
 

Now, we can solve the problem by linear programming.

How do we minimising $\mu_m(s, t)$ when the terrains are *not* aligned?



Now, the maximum distance can occur at a vertex of f or g, or at a crossing of their edges.



Same algorithm works, but now complexity is $O(n^2)$ because of worst-case overlay.

Bla bla bla bla bla, bla bla bla random sampling bla bla bla, bla bla bla bla bla bla violated constraints bla bla bla bla bla. Bla two-level heriditary segment trees bla bla bla bla bla bla, bla bla bla bla bla bla, bla bla bla bla bla bla Plücker points bla bla bla? Bla bla bla bla bla bla bla bla 5-dimensional projective space bla bla bla bla bla bla: bla bla trade-off bla bla bla. Bla bla, bla bla bla bla bla bla bla bipartite clique decomposition bla bla bla bla bla bla bla bla bla bla bla bla bla bla, bla bla bla bla bla bla bla bla halfspace reporting queries bla!

This can be improved to $O^*(n^{4/3})$ by using some tricks.

Let’s conclude.

Let's conclude.

We want to find linear relationships between functions on the same domain.

Let's conclude.

We want to find linear relationships between functions on the same domain.

We model this as minimising the vertical distance between two TINs in $\mathbb{R}^3$.

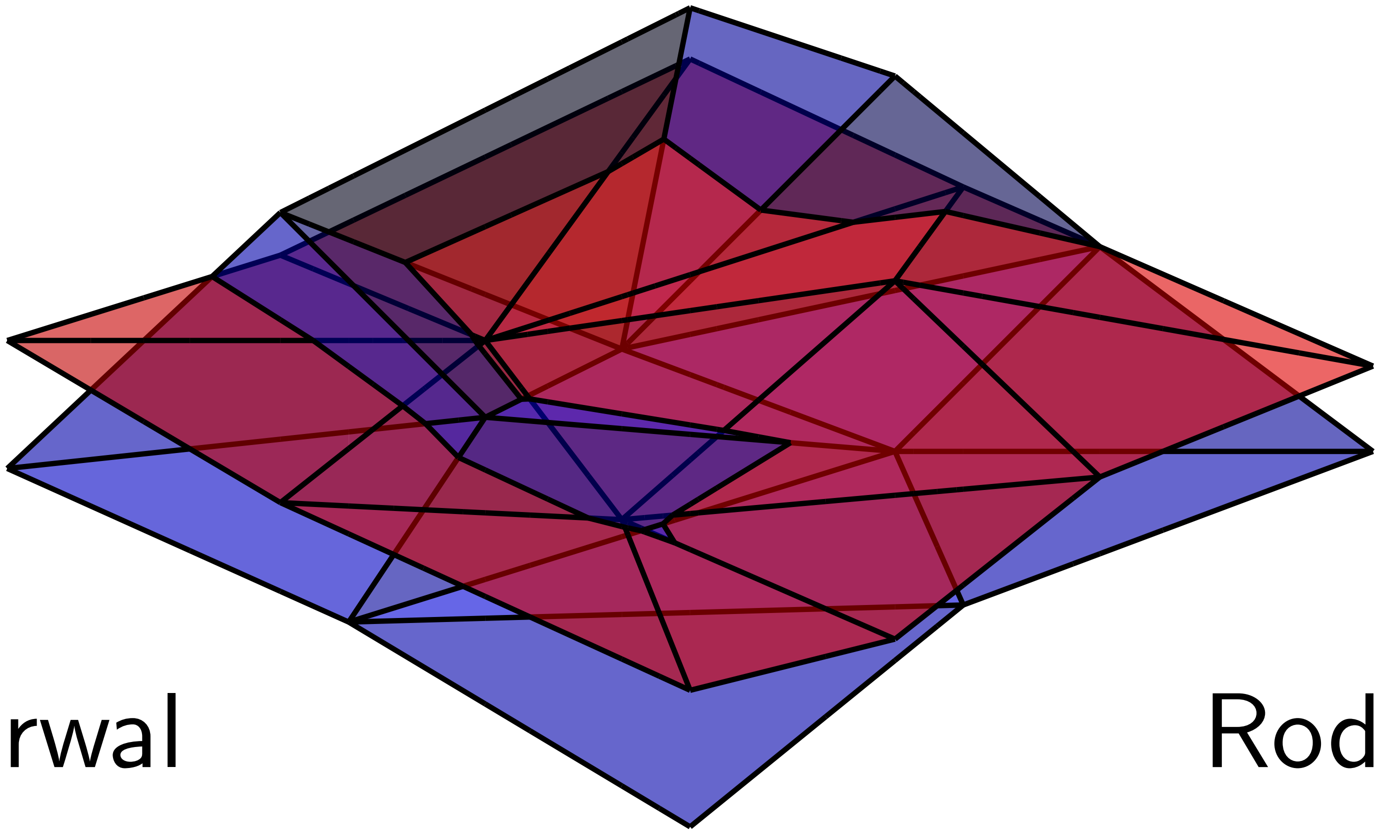
Let's conclude.

We want to find linear relationships between functions on the same domain.

We model this as minimising the vertical distance between two TINs in $\mathbb{R}^3$.

We can efficiently minimise the maximum distance. The average distance requires more work.

Thank you!



Pankaj Agarwal

Rodrigo Silveira

Boris Aronov

Maarten Löffler

Marc van Kreveld