Robust algorithm for the intersection of simplices

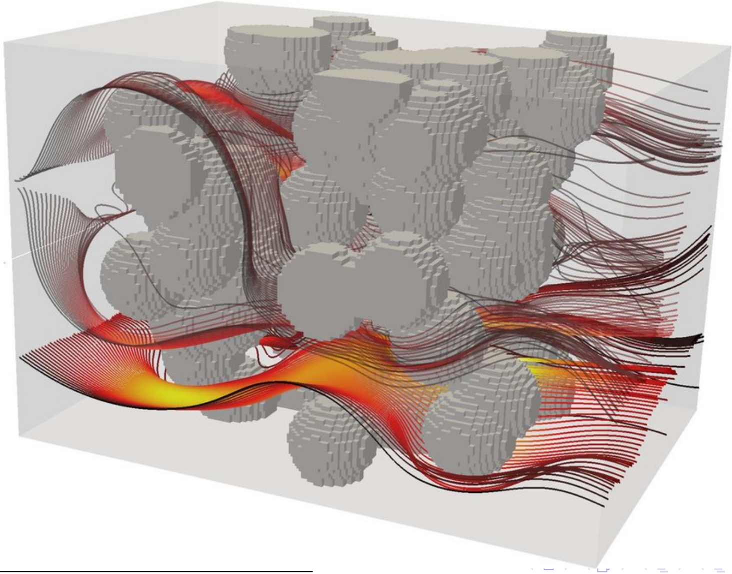
Conor McCoid [1], Martin J. Gander [2]

[1] Université Laval, [2] Université de Genève

03.12.2022

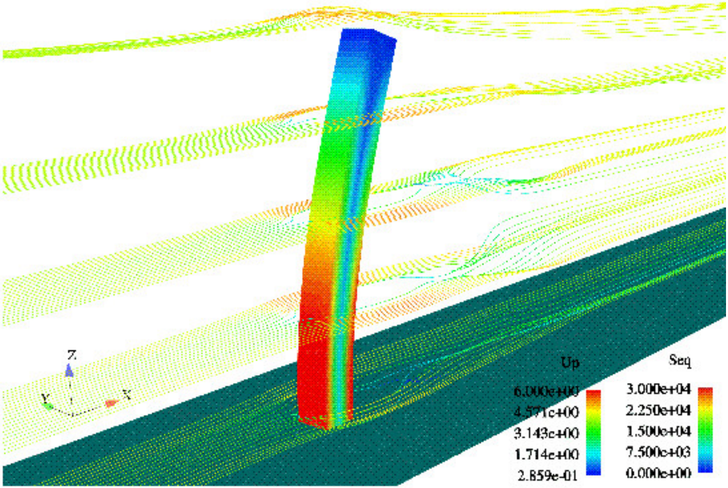
Why multiple grids?

Why multiple grids?



1

Why multiple grids?



[1]

[1] **Sasongko2009**

What can go wrong?

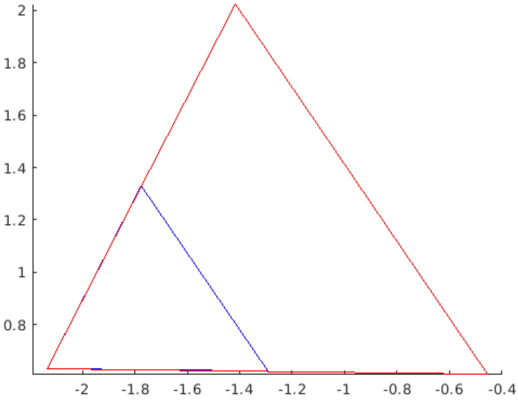


Figure: An obvious problem

What can go wrong?

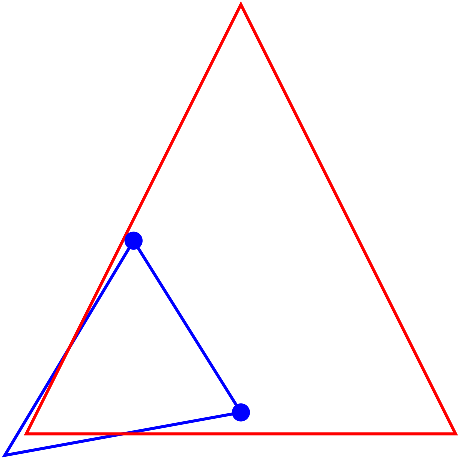


Figure: Blue vertices in red triangle

What can go wrong?

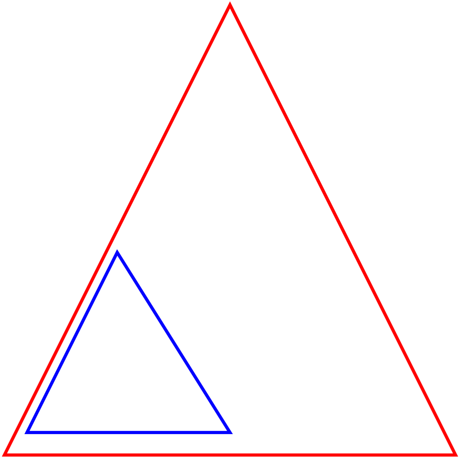


Figure: Red vertices in blue triangle

What can go wrong?

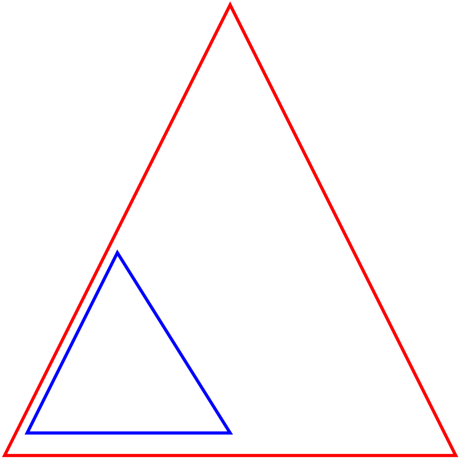


Figure: Edge intersections

What can go wrong?

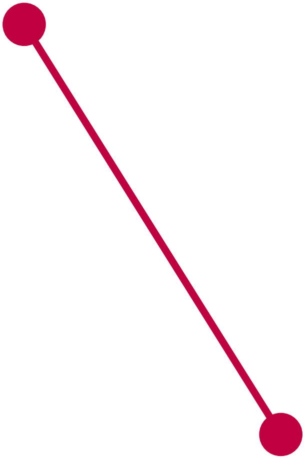


Figure: Final result - no intersection!

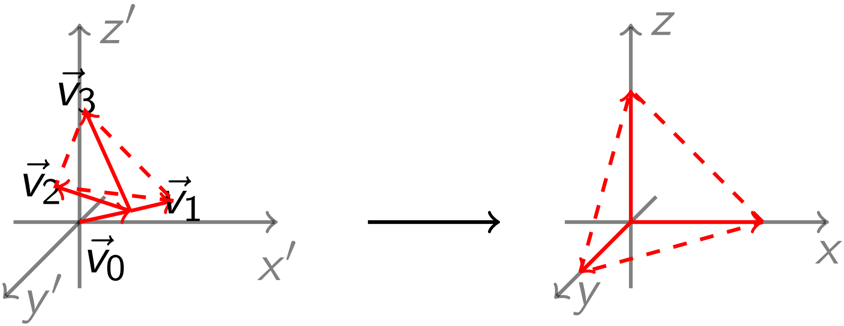
What to do about it?

We need consistency!

Parsimony: the principle of using the fewest resources to solve a problem.

If an algorithm is parsimonious it is self-consistent, meaning the result represents a possible accurate outcome, even if it's inaccurate for the given problem.

Change of coordinates



General dimension, $\mathbb{R}^n$:

$$\begin{bmatrix} \vec{v}_1 & \dots & \vec{v}_n \end{bmatrix} \begin{bmatrix} \vec{x}_0 & \dots & \vec{x}_n \end{bmatrix} = \begin{bmatrix} \vec{u}_0 - \vec{v}_0 & \dots & \vec{u}_n - \vec{v}_0 \end{bmatrix}$$

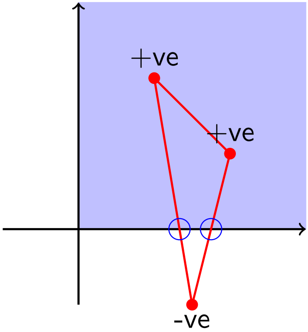
We call the reference simplex Y and the general one X.

Sectioning by (hyper)planes

Avoid degenerate cases: binary-valued sign function

$$\text{sign}(p) = \begin{cases} 1 & p \geq 0, \\ 0 & p < 0, \end{cases}$$

We only calculate an intersection between two points if they have different signs in a given direction



Sectioning by (hyper)planes

Definition (Simplex)

A simplex in $\mathbb{R}^n$ is the intersection of $n+1$ half-spaces bounded by $n+1$ hyperplanes of codimension 1.

Those hyperplanes are:

$$P_i = \{\vec{x} \in \mathbb{R}^n \mid \vec{x} \cdot \vec{e}_i = 0\}$$

In 2D, they are the lines $x = 0$, $y = 0$ and $x + y = 1$

In 3D, the planes $x = 0$, $y = 0$, $z = 0$ and $x + y + z = 1$

Sectioning by (hyper)planes

Small trick: add the coordinate $\vec{e}_0$ such that

$$\vec{x} \cdot \vec{e}_0 = 1 - \sum_{i=1}^{n} \vec{x} \cdot \vec{e}_i$$

Now the $n+1$ coordinates are barycentric with respect to one of the simplices

Sectioning by (hyper)planes

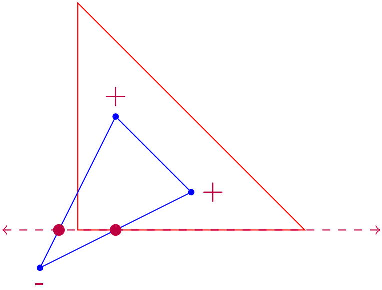
Take a vertex ($\vec{x}_j$) and check its sign ($\text{sign}(\vec{x}_j \cdot \vec{e}_i)$) for all hyperplanes; if it's positive for all of them then $\vec{x}_j$ lies in the reference simplex Y:

$$\chi_Y(\vec{x}_j) = \prod_{i=0}^{n} \text{sign}(\vec{x}_j \cdot \vec{e}_i)$$

Now we can relate the number of vertices inside Y with the number of intersections

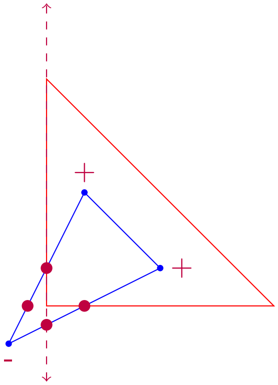
Do-over on opening example

Let's look at the original 2D example that failed



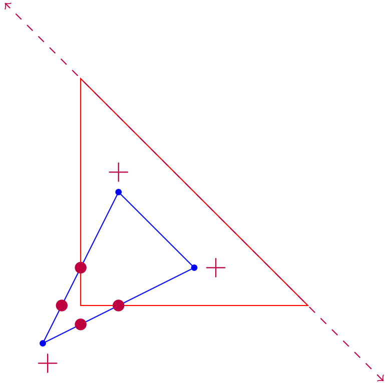
Do-over on opening example

Repeat for the other hyperplanes



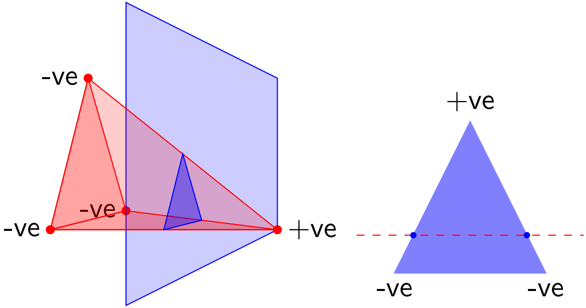
Do-over on opening example

Repeat for the other hyperplanes



Higher dimensional sectioning

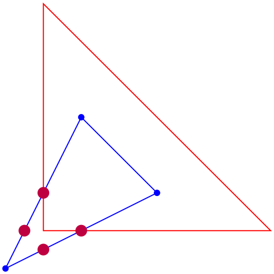
In higher dimensions, the process repeats



The embedded triangle is sectioned by other hyperplanes, with the intersections taking on the role of vertices

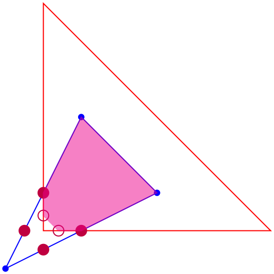
Intersections of edges

Returning to the 2D example:



What were to happen if there was an error in calculating the intersections?

Intersections of edges



How do we prevent this from happening?

Intersections of edges

Intersection along the line $y = 0$:

$$q_y^{\{0,1\}} = \frac{x_0 y_1 - x_1 y_0}{y_1 - y_0}$$

Intersection along the line $x = 0$:

$$q_x^{\{0,1\}} = \frac{x_0 y_1 - x_1 y_0}{x_0 - x_1}$$

Intersections of edges

Intersection along the line $y = 0$:

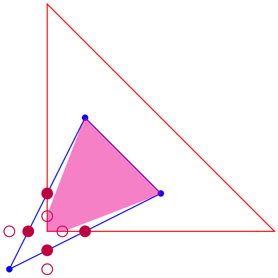
$$q_y^{\{0,1\}} = \frac{x_0 y_1 - x_1 y_0}{y_1 - y_0}$$

Intersection along the line $x = 0$:

$$q_x^{\{0,1\}} = \frac{x_0 y_1 - x_1 y_0}{x_0 - x_1}$$

Same numerator!

Intersections of edges



Intersection of k-faces

Intersections for k-faces, after sectioning by k hyperplanes:

$$\vec{q}^{J}_{\Gamma} \cdot \vec{e}_{i_0} = \frac{\begin{vmatrix} \vec{x}_{j_0} \cdot \vec{e}_{i_0} & \vec{x}_{j_0} \cdot \vec{e}_{i_1} & \dots & \vec{x}_{j_0} \cdot \vec{e}_{i_k} \\ \vdots & \vdots & & \vdots \\ \vec{x}_{j_k} \cdot \vec{e}_{i_0} & \vec{x}_{j_k} \cdot \vec{e}_{i_1} & \dots & \vec{x}_{j_k} \cdot \vec{e}_{i_k} \end{vmatrix}}{\begin{vmatrix} 1 & \vec{x}_{j_0} \cdot \vec{e}_{i_1} & \dots & \vec{x}_{j_0} \cdot \vec{e}_{i_k} \\ \vdots & \vdots & & \vdots \\ 1 & \vec{x}_{j_k} \cdot \vec{e}_{i_1} & \dots & \vec{x}_{j_k} \cdot \vec{e}_{i_k} \end{vmatrix}}, \quad \begin{cases} J = \{j_0, \dots, j_k\}, \\ \Gamma = \{i_1, \dots, i_k\} \end{cases}$$

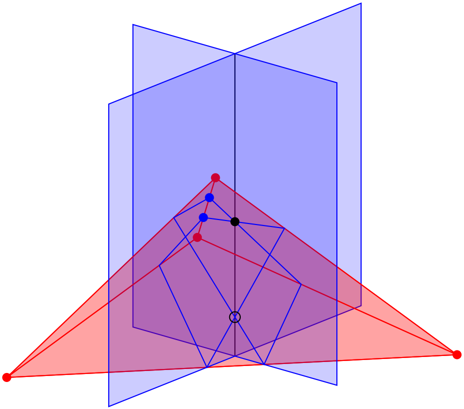
Intersection of k-faces

Swap around the columns for the other intersections of this k-face:

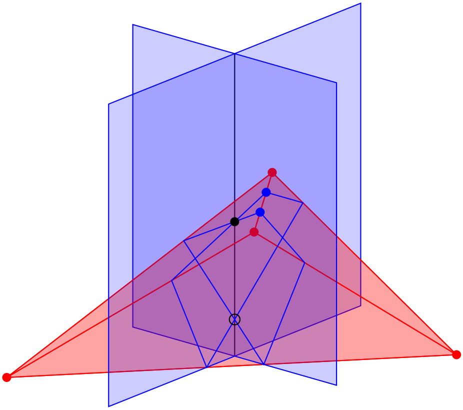
$$\vec{q}^{J}_{\Gamma_1} \cdot \vec{e}_{i_1} = \frac{\begin{vmatrix} \vec{x}_{j_0} \cdot \vec{e}_{i_1} & \vec{x}_{j_0} \cdot \vec{e}_{i_0} & \dots & \vec{x}_{j_0} \cdot \vec{e}_{i_k} \\ \vdots & \vdots & & \vdots \\ \vec{x}_{j_k} \cdot \vec{e}_{i_1} & \vec{x}_{j_k} \cdot \vec{e}_{i_0} & \dots & \vec{x}_{j_k} \cdot \vec{e}_{i_k} \end{vmatrix}}{\begin{vmatrix} 1 & \vec{x}_{j_0} \cdot \vec{e}_{i_0} & \dots & \vec{x}_{j_0} \cdot \vec{e}_{i_k} \\ \vdots & \vdots & & \vdots \\ 1 & \vec{x}_{j_k} \cdot \vec{e}_{i_0} & \dots & \vec{x}_{j_k} \cdot \vec{e}_{i_k} \end{vmatrix}}, \quad \begin{cases} J = \{j_0, \dots, j_k\}, \\ \Gamma_1 = \{i_0, i_2, \dots, i_k\} \end{cases}$$

There's $k+1$ intersections (same k-face of X, different hyperplanes of Y) that share this numerator

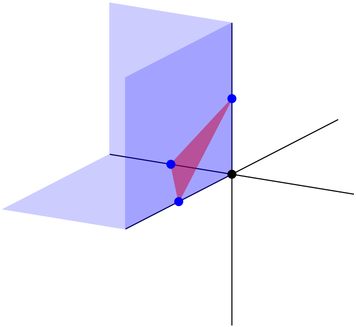
Intersection of k-faces



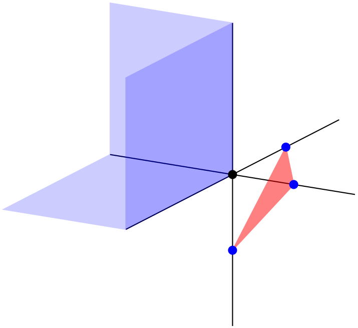
Intersection of k-faces



Intersection of k-faces

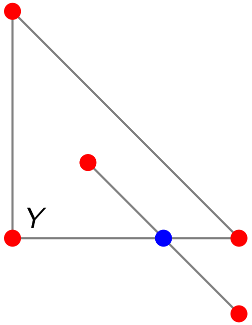


Intersection of k-faces



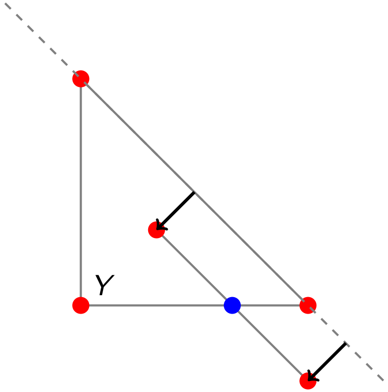
Partial intersecting k-faces

What if there aren't $k+1$ intersections between the k-face and the hyperplanes of Y?



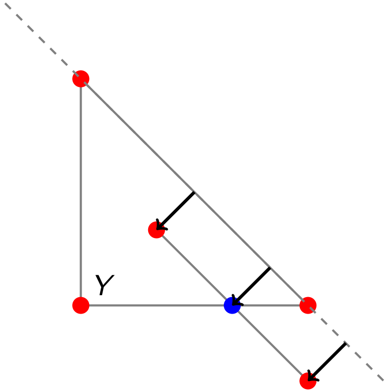
Partial intersecting k-faces

Then there's at least one hyperplane that does not section the k-face, so all intersections of the previous generation all have the same sign



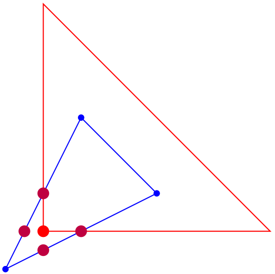
Partial intersecting k-faces

This sign transfers to the next generation



Y vertices in X

Vertices of Y that lie in X sit between intersections of $(n-1)$-faces (called facets in geometry)



Take any line of Y that passes through the vertex and see if the vertex lies between two intersections

Conclusions

Robustness is achieved through parsimony

By making sure all calculations agree, we ensure the result corresponds to a possible intersection

Any inaccuracy will then come from the individual calculations, and not the way in which the algorithm proceeds

Future works

Can this be applied to convex polytopes in general? What about concave polytopes? Disconnected polytopes?

What about curved shapes? Spheres, ie. hyperbolic geometries?

We can apply parsimony to any algorithm; what other algorithms could benefit from similar approaches?