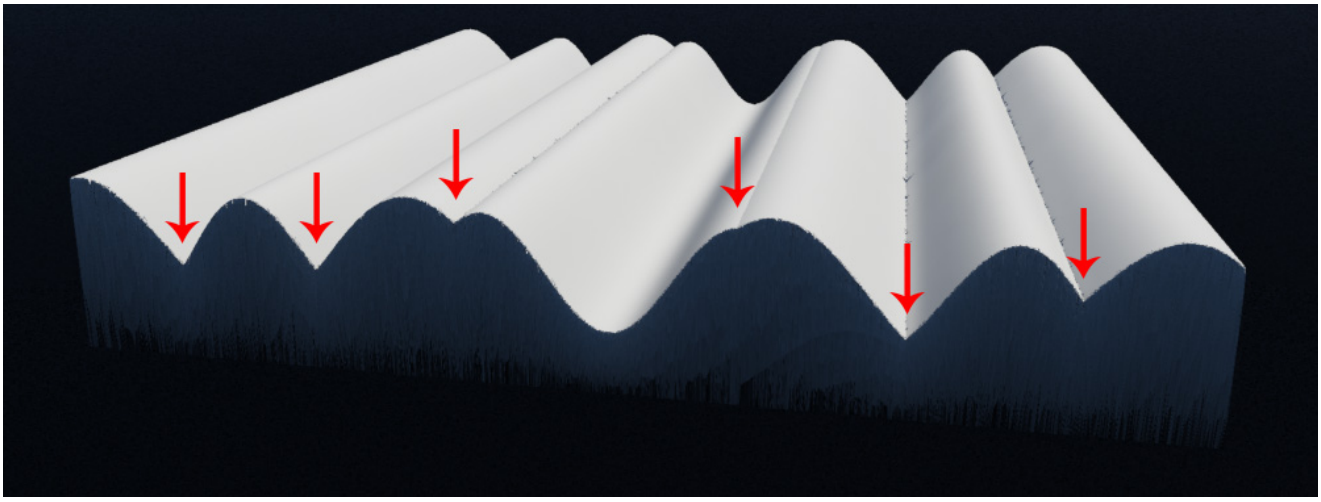
Smooth combiner
with Terragen 2 / Terragen 3

A tutorial by Laurent Avenel (Wiwine)

The smooth combiner is a little module that can be useful when we have to work with function-generated terrains. In some situations we may work with multiple functions and try to combine them, keeping the max values for each function. For this we will use a « conditional scalar » with the parameter « greater than ».

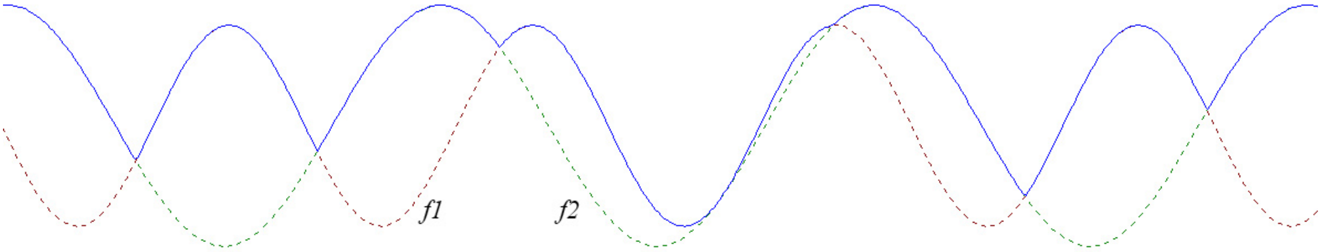
In that case, when the curves intersect, we will obtain some corners (inverted peaks) in the low parts, that are not very pretty. Here is an example of this situation (with two trigonometric curves) :



This is the result of the combination of the following functions $f1$ and $f2$:

$f1 = 12 \cdot \sin(\frac{x}{10} + 5)$ and $f2 = 10 \cdot \sin(\frac{x}{7} - 8)$ *(but any function can do the trick)*

The rendered curve is the $\max(f1, f2)$ restrained to a square area of 200 × 200 (used to see the shape of the curve). Here are the same functions in a classical 2D graph. $f1$ and $f2$ are the dotted lines, $\max(f1, f2)$ is in blue :

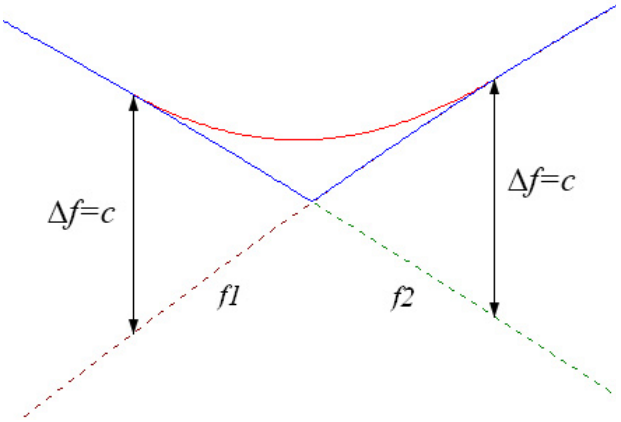


The idea is to add a complementary function around the parts where $f1$ and $f2$ intersect, to create a fillet.
This complementary function will depend on $f1$ and $f2$, and more precisely on the difference (absolute value) between the two functions : $\Delta f = |f1 - f2|$
$\Delta f = |f1 - f2|$ is easy to obtain : we just need a « difference scalar » shader for it.

When the two functions come close, the value of Δf decreases, and when $f1$ and $f2$ intersect, $\Delta f = 0$
We will add the complementary function to the main curve when Δf is below a specific value.

So we have to define a parameter that will be the max value of Δf for which we will use the complementary function.
Let's call this parameter « *c* ».

Here is a closer view on our previous functions, with the complementary function that we want to obtain :



As before, $\max(f1, f2)$ is in blue, and the complementary function is in red.

This new function is not far from a parabolic curve, depending on the value of $\Delta f = |f1 - f2|$:

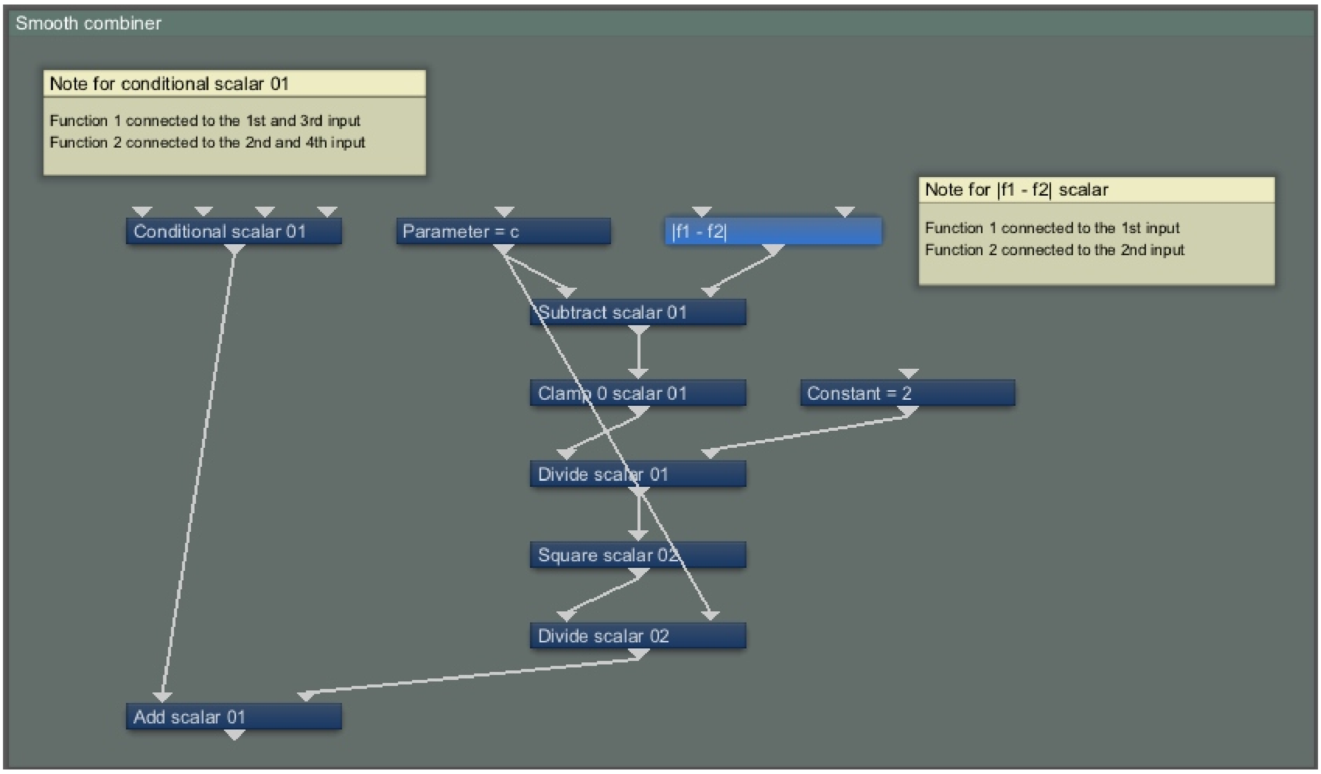
- if $\Delta f = 0$ (intersection of lines), this new function is at its max.
- if $\Delta f = c$ (left and right limits of the smoothing interval), this new function equals 0.
- if $\Delta f > c$ (beyond the smoothing interval, both sides), we don't use the new function.

The formula for this complementary function is : $\frac{1}{c} \cdot \left(\frac{c - \Delta f}{2} \right)^2$ with $\Delta f \leq c$

Then the whole smoothed curve will be : $\max(f1, f2) + \frac{1}{c} \cdot \left(\frac{c - \Delta f}{2} \right)^2$

This method works with almost any kind of functions, and even with the « power fractal » shaders if we use the high and low colours for the displacement values.

The node tree for this formula looks like this :



Note for conditional scalar 01

Function 1 connected to the 1st and 3rd input
Function 2 connected to the 2nd and 4th input

Note for |f1 - f2| scalar

Function 1 connected to the 1st input
Function 2 connected to the 2nd input

The « clamp 0 scalar » is here to restrict the complementary function to the interval where $\Delta f \leq c$

All we have to do is :

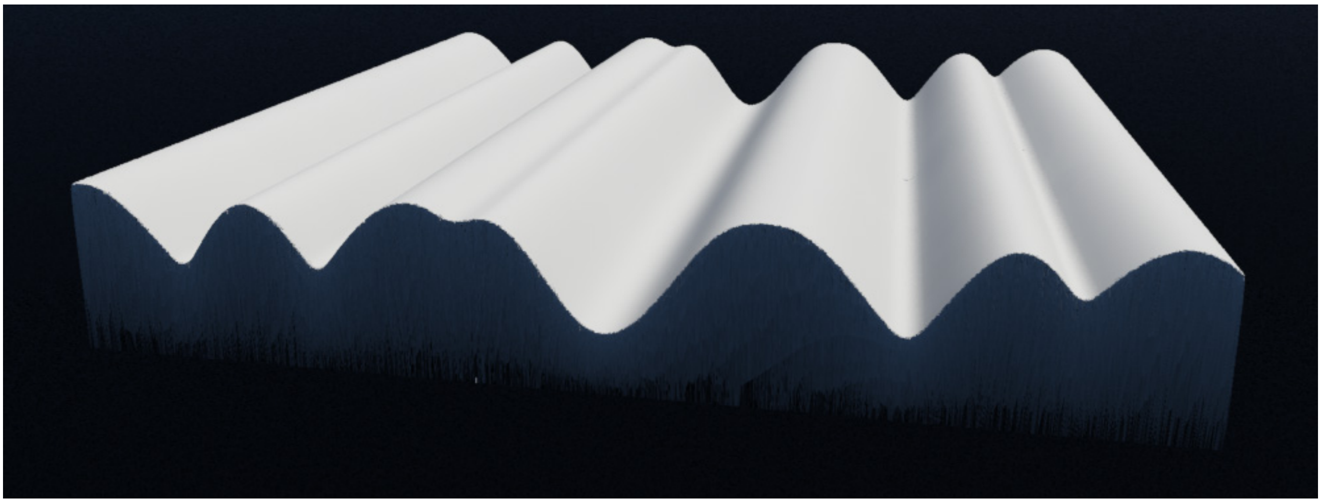
- connect the first function to the « conditional scalar » (1st and 3rd inputs) and to the « $|f1 - f2|$ » scalar (1st input)
- connect the second function to the « conditional scalar » (2nd and 4th inputs) and to the « $|f1 - f2|$ » scalar (2nd input)
- connect the output of the final « add scalar » to the next part of your process (a displacement shader for example)
- adjust the value of the c parameter

The value for the `c` parameter totally depends :

- on the scale and shape of the curves generated by the functions
- on the size of the smoothing curve that you want.

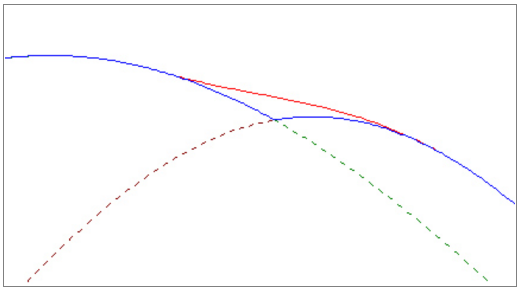
So it's impossible to give a generic value ; you will have to test the process before choosing the adequate value of c.

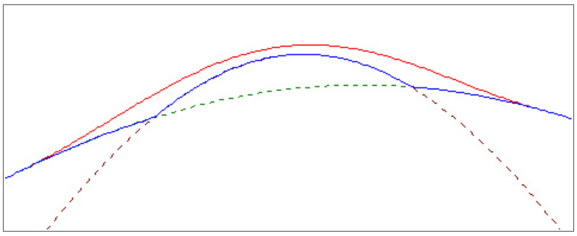
Here is the new function applied to the curves $f1$ and $f2$:

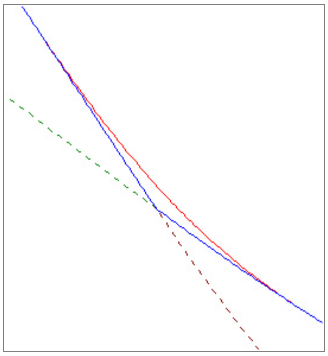


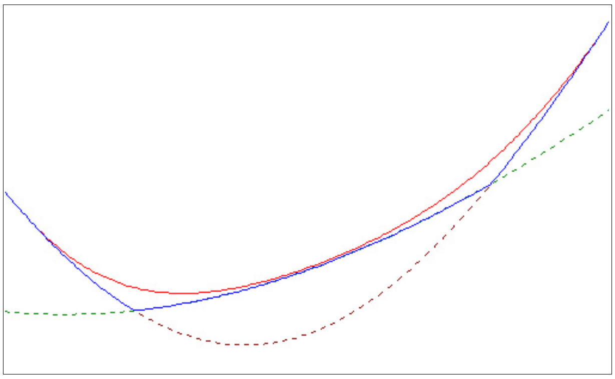
The resulting curve is smoothed : there's no more corner. As it's used for a terrain generation, this can be seen as filling by erosion materials (sand, gravel, mud, snow…).

Depending on the way the curves $f1$ and $f2$ intersect, the complementary function sometimes gives some curious shapes, but in most cases it works well (even if the curves are close without intersection).









To finish, there is a little .gif animation in the files pack, that shows the effect of this module with two « power fractal » shaders used to generate a terrain. These two shaders are combined with a « conditional scalar », then the complementary function is applied.

The animation shows the progressive effect when the value of the *c* parameter increases from 0 (no smoothing) to 18 : the intersection lines are gradually erased. (There are also some faint lines that remain unaffected by the smoothing, but they have nothing to do with the PF combination.)

In the files pack :

- smooth-combiner.tgc : the clip file
- smooth-combiner-anim.gif : the animation
- smooth-combiner-example.tgd : the file used to create the pictures for the animation