Subject: Re: Capture division by zero
Posted by mirek on Mon, 02 Nov 2020 09:31:01 GMT
View Forum Message <> Reply to Message

koldo wrote on Thu, 29 October 2020 14:19mirek wrote on Wed, 28 October 2020 14:04I am also pretty interested what will you do when you get exception in eigen. I am not that advanced in math, but I am pretty sure that e.g. matrix inversion can produce a lot of divide by zero for degenerate matrix and I am also pretty sure that eigen follows NaN model like every other mature library (if for nothing else, then for performance reasons).

Are you going to compute determinant before each call to inversion?

Mirek
I am well aware of operations that could be bad conditioned :). I use a lot matrix inversion.
In fact STEM4U includes some tools to handle infinite precision numbers :lol: (well, but as expected, very inefficiently)
If you ask me if I prefer to search for NaNs after every matrix algebra operation, or catching an exception, I will choose the second.

Now actually catching the exception as part of program logic in release mode (as oppposed to just stopping on NaN in debug), that is completely different thing! I could agree that would be a good solution, except for one nasty tiny detail: you cannot officially catch FP exceptions in C++ (see e.g. https://stackoverflow.com/questions/38221471/how-to-convert- sigfpe-into-a-c-exception). It might be possible on specific platforms / compilers, but I would think twice before making your code reliant on it

All you can realistically do in the handler is to set some per-thread and/or global "fp-error" flag. That frankly does sound quite similar to NaN...

All that said, activating FP exception might have its uses, e.g. when you are getting NaN out of calculation for unknown reasons and you want to track it down.

I just do not think that it is the only valid way. And in my humble experience, insisting that your code never produces NaNs creates much more mess than it solves.

Mirek