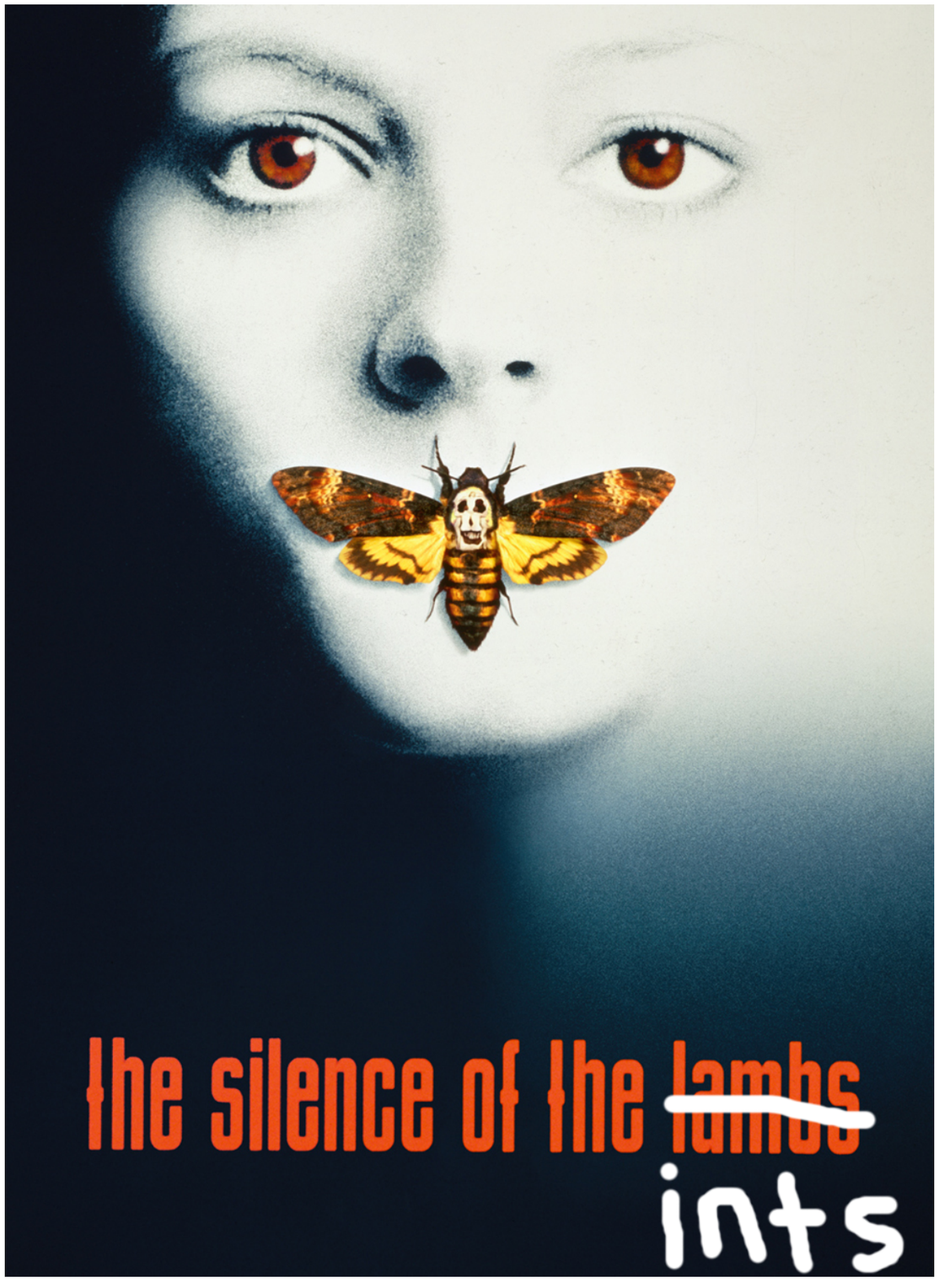
Thomas Kober
@tttthomasssss



PyData Edinburgh
4th Apr 2019
(1554398100)

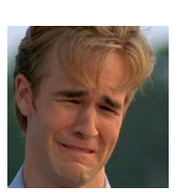
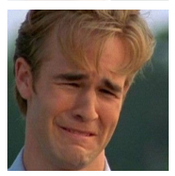
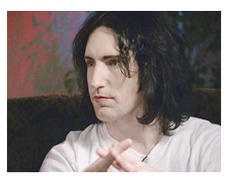
Whats this all about?

- The world is full of **SILENTLY OVERFLOWING** integers
- In my last lightning talk, I've shown that **SILENT INTEGER OVERFLOWS** can happen in `numpy` and `scipy`
- In this, we'll have a look at some other commonly used libraries

Bring on the failing Code

- *(Note to self: open the Jupyter Notebook now)*

OK, that looks fairly bad

- Is this a python-specific thing?
- Not really, I've tried the same with a few other languages:
  - Octave  *It fails*
  - Matlab  *It fails*
  - Julia  *It works, no silent integer overflows in Julia!!!*
  - R  *(I don't know, I didn't try)*

So, why??

- There are two important questions to ask
  - *Why does it* FAIL *for* ***integers****?*
  - *Why does it* NOT *fail for* ***floats****?*
- Why it doesn't fail for floats
  - Floating point errors are checked at the hardware level, i.e. the FPU sets a flag whenever something goes wrong. Errors are propagated upwards. This is very efficient.
- Why it fails for integers
  - Integer overflows are NOT checked at the hardware level. Any client code needs to implement its own checks. In the case of `numpy`, the issue is known since at least 2009, but there was a deliberate decision to NOT check integer overflows in arrays for performance reasons (see here and here).