

Abstract. Ordinary category theory is too rigid to handle equivalence correctly. In algebraic topology, we study maps up to homotopy and \simeq , but the category \mathbf{hTop}_* does not admit pushouts, because we have forgotten how two maps are equivalent. The story is similar in homological algebra: $D(R)$ does not admit colimits, so Verdier invented triangulated categories as a "1-categorical patch". Yet, it doesn't capture the true story; the cone-construction cannot be made functorial. We have lost too much information by throwing away how two chain-complexes are equivalent. This is one problem that ∞ -categories fix. Triangulated categories will be replaced by *stable* ∞ -categories, which are easier to define, and from which we see/motivate the axioms of a triangulated category - remarkably the evil octahedral axiom will turn out to be just some iterated pushouts given two maps.

Other applications in algebra include formalising Grothendieck's six functors $(f^*, f_*, f_!, f^!, \otimes, \underline{\mathbf{Hom}})$ which we will see an instance of later during the camp.

This class is meant to give you an idea of what goes into setting up the language of ∞ -categories, how usual concept from ordinary category theory incarnate in ∞ -categories, and how ∞ -categories can be applied in the "real" world. Note that the theory of ∞ -categories is way too vast to cover in just a couple of days so expect a ton of black boxes.

Prerequisites. With that out of the way,

- Go to Igor's class on simplicial methods.
- 1-category theory like Yoneda, adjunction, and universal property of presheaf category. This can be obtained during the first class.
- Not hard prerequisite, but knowing a bit of model categories will make your life easier.
- Not hard prerequisite, but some familiarity with the usual $D(R)$ (this will only be relevant at the very end).

Literature.

- Lurie: Higher Topos Theory, Higher Algebra, Kerodon.net
- Markus Land: Introduction to Infinity-categories
- Ferdinand Wagner: ∞ -Categories in Topology
- ...