Bing

Annotations in Python can indeed provide type information and are useful for static type checkers and Integrated Development Environments (IDEs). However, the `typing` module offers more comprehensive and precise type hinting compared to basic annotations.

Here's why:

1. **Complex Types**: The `typing` module allows you to express complex types, such as `List[int]` or `Dict[str, int]`, which cannot be expressed with basic annotations.

2. **Standardization**: The `typing` module is a standardized way of providing type information, which means tools and IDEs are more likely to support it.

3. **Stringized Annotations**: Python 3.10 introduced a new function, `inspect.get_annotations()`, which can "un-stringize" stringized annotations. This is particularly useful when dealing with forward references, or when the type is not yet defined at the time the annotation is read.

4. **Better Tooling Support**: Static type checkers like `mypy`, `pyright`, `pytypes`, or `pyre` can leverage type hints for static type-checking and provide hints or warnings when these types are used inconsistently.

So, while annotations can be helpful, the `typing` module provides a more robust and precise way of hinting types in Python.