# Copilot: Python long calculation

**Prompt**

What kind of technique are supported in python for heavy, long calculation, not overloading main thread?

When you're dealing with **heavy or long-running calculations** in Python and want to avoid freezing or overloading the main thread—especially in web apps or GUIs—there are several solid techniques you can use:

**🧵 1. Threading**

* Good for I/O-bound tasks (like file or network operations).
* Not ideal for CPU-heavy work due to Python’s **Global Interpreter Lock (GIL)**.
* Use threading.Thread or concurrent.futures.ThreadPoolExecutor.

**🧮 2. Multiprocessing**

* Best for **CPU-bound** tasks.
* Spawns separate processes, bypassing the GIL.
* Use multiprocessing.Process or concurrent.futures.ProcessPoolExecutor.

**⚙️ 3. AsyncIO**

* Great for **asynchronous I/O-bound** tasks (e.g., many API calls).
* Doesn’t help with CPU-heavy work unless combined with multiprocessing.
* Use async def, await, and asyncio.run().

**🧠 4. Job Queues / Task Schedulers**

* For background tasks in web apps (like Django), use:
  + **Celery** (with Redis or RabbitMQ)
  + **Django-Q**
* These let you offload long tasks to workers.

**🚀 5. NumPy / Numba / Cython**

* For raw speed in numerical tasks:
  + **NumPy**: Vectorized operations.
  + **Numba**: JIT compiler for numeric Python code.
  + **Cython**: Compile Python to C for performance.

**🧪 6. Offload to GPU**

* Use **CuPy** or **PyTorch** for GPU-accelerated computation if you’re working with large matrices or ML.

If you tell me more about the kind of calculation you're doing—like data processing, simulations, or ML—I can help you pick the best combo. Want to try a multiprocessing example?