GEO大型series matrix读取：先选对数据，再分块检查

先判断GEO文件是否适合当前分析，再逐块读取series matrix，保留样本顺序和缺失值，并检查平台、行数与输入完整性。

GEO 的 series matrix 适合快速取得提交者整理过的表达矩阵，但“大”不等于“可以直接用于任意分析”。同一 GSE 可能对应多个 GPL 平台；芯片探针强度、已标准化表达值和 RNA-seq 计数也不是同一种量。本文只演示读取一个结构规范、单平台、数值型 series matrix，并把表达行分块写出，降低一次性把整张矩阵装入内存的压力。

先确认下载对象适合问题

在 GEO 系列页记录 GSE、样本 GSM、平台 GPL、物种、组织和实验分组，再看 series matrix 文件说明及补充文件。若目标是 RNA-seq 原始 reads，应找 FASTQ/SRA；若需要作者提供的原始 counts 或特定处理结果，应确认补充文件、GEO 预计算定量或作者说明，而不要把 series matrix 的数值默认当作 raw counts。芯片数据还要查看 GPL 探针注释、是否已 log 转换及归一化方法。

一个 GSE 可能有多个平台，对应多个矩阵或 GEOquery 返回的多个平台对象。不同 GPL 的探针 ID 和覆盖范围不能仅因行数相近就纵向拼接。先选择与研究问题匹配的平台；如确需跨平台比较，需另外制定探针到基因的映射和批次处理方案。下面的脚本遇到多个表会停止，要求先拆开核对。

读取前检查	需要记录	发现不一致时
样本列	表头 GSM 与 GEO 页面顺序	先找出缺样本或重复列，不要静默重排
平台	GPL 编号与探针注释版本	不同平台分开处理
数值	单位、log/归一化及缺失编码	回到提交说明或补充文件
文件	下载地址、日期、大小和 SHA256	重新下载并比对完整性

分块读取并保留原始顺序

先安装 Python 3。把下列脚本保存为 split_geo_matrix.py，并将输入文件设为官方 series matrix 的 gzip 压缩文件。脚本只接受单个 ID_REF 表，按原顺序输出 TSV 分块，写入原始样本列名、特征 ID、行号和元数据；数字仅做有限值检查，不进行归一化、探针汇总或缺失值填补。输出目录须不存在，失败时不会生成成功标记。

```
import csv, gzip, hashlib, json, math, sys
from pathlib import Path
def sha256_file(path):
    h = hashlib.sha256()
    with path.open("rb") as f:
        for chunk in iter(lambda: f.read(8*1024*1024), b""): h.update(chunk)
    return h.hexdigest()
src, dest = Path(sys.argv[1]), Path(sys.argv[2])
dest.mkdir()  # ■■■■■■■■■■■■■■■
meta, blocks, samples = [], [], None
missing = rows = 0
with gzip.open(src, "rt", encoding="utf-8", newline="") as f:
    r = csv.reader(f, delimiter="\t")
    for line in r:
        if line and line[0] == "!series_matrix_table_begin": break
        if line and line[0].startswith(("!Series_", "!Sample_", "!Platform_")):
            meta.append(line)
    else: raise ValueError("■■■ matrix table ■■■■")
    header = next(r)
    if len(header) < 2 or header[0] != "ID_REF": raise ValueError("■■■ ID_REF")
    samples = header[1:]
    if any(not x for x in samples) or len(set(samples)) != len(samples):
        raise ValueError("■■■■■■■■■")
    gsm = [x[1:] for x in samples]
    accession = [x[1:] for x in meta if x[0] == "!Sample_geo_accession"]
    if accession and accession[0] != gsm: raise ValueError("GSM■■■■■■■■■")
    fh = writer = buf = None
    for row in r:
        if row and row[0] == "!series_matrix_table_end": break
        if len(row) != len(header) or not row[0]: raise ValueError("■■■■■■■ID■■")
        for value in row[1:]:
            if value.strip().lower() in {"", "na", "nan", "null"}: missing += 1
            elif not math.isfinite(float(value)): raise ValueError("■■■■■■■■")
        if rows % 2000 == 0:
            if fh: fh.close()
            name = f"block-{len(blocks)+1:05d}.tsv"
            fh = (dest/name).open("w", encoding="utf-8", newline="")
            writer = csv.writer(fh, delimiter="\t", lineterminator="\n")
            writer.writerow(["SOURCE_ROW", *header])
            blocks.append(name)
        writer.writerow([rows+1, *row]); rows += 1
    else: raise ValueError("■■■ matrix table ■■■■")
    if fh: fh.close()
    for row in r:
        if row and row[0] == "!series_matrix_table_begin":
            raise ValueError("■■■■■■■■■■■■■■■■■■■")
if rows == 0: raise ValueError("■■■■■■■■")
h = sha256_file(src)
report = {"source": str(src), "sha256": h, "samples": samples,
          "features": rows, "missing_values": missing, "blocks": blocks,
          "metadata": meta}
(dest/"manifest.json").write_text(json.dumps(report, ensure_ascii=False, indent=2),
encoding="utf-8")
(dest/"STRUCTURE_CHECK_OK").write_text("Rows and sample columns checked; values not
normalized.\n", encoding="utf-8")
print(f"features={rows} samples={len(samples)} missing={missing} sha256={h}")
```

怎样读检查结果

运行示例：python split_geo_matrix.py GSE123456_series_matrix.txt.gz GSE123456_blocks。检查 manifest.json 中的输入 SHA256、样本顺序、特征行数、缺失数和分块清单；每块 SOURCE_ROW 应从 1 连续递增，首尾列数及 ID_REF 要能和原表对应。脚本按 8 MiB 分块计算 SHA256，不把整个压缩包载入内存。结构检查只说明表格可读且列数一致，不证明表达值尺度、样本分组或统计模型正确。缺失计数对应原文件的空值/NA/null 标记，不能未经分析计划就填成 0。

进入差异分析前，再单独整理样本表：用 GSM accession 作唯一键，保留 GEO 原始标题和 characteristics，同时把清洗后的组织、处理组、时间点和重复号放在新列。不要只凭标题自动推断组别；遇到缺失或相互冲突的标签，回到原始样本页和论文方法核对。表达矩阵列名必须能与样本表一一对应，过滤样本时同步更新两边并记录原因。

对芯片数据，先按 GPL 注释确认一行代表探针还是基因；多个探针对应同一基因时，汇总规则会影响结果，应在分析方案中明确。对 RNA-seq，series matrix 通常是提交者处理后的结果；若研究需要整数原始 counts，应另找明确标注的 counts 文件或测序数据处理流程。不要对已 log 的值再次做 log，也不要因矩阵列名看起来像样本编号就假定其处理方式已知。

参考资料

GEOquery 官方教程：GSE、平台与补充文件 官方文档
GEOquery series matrix 解析实现 官方文档
Python csv 标准库文档 官方文档