上海交通大学 2023 ~ 2024 学年度 应用统计公众号文章项目

软件运行效率对比

2024 - 01 - 17 编制

编者院系：**数学科学学院**　指导教师：**王 成**　共同完成者：**李昌骏　刘若禺　邵毅诚**

(共同完成者依照姓氏拼音字母顺序排序)

前　言

本项目受到 2023 学年秋季开设的《统计软件与算法》课程的启发, 对于 R 语言当中可采用的加速库 BLAS, OpenBLAS, 以及 Python, Matlab 语言, 对于多个数学, 统计领域相关运算任务进行软件运行效率的对比.

第一部分　引　例

在当今信息时代, 数据分析已经成为各行各业中不可或缺的一部分. 在金融、互联网、生物医学、市场营销等众多研究领域, 数据分析为决策提供了可视化的数据支撑材料. 然而, 在庞大的数据海洋中航行, 并从中提取有用的信息涉及到诸多复杂的统计主题.

在本文中, 我们将聚焦于三个大类的统计任务主题：生成大数据、回归分析以及高维数据主成分分析. 这些主题代表了实际应用中常见的数据分析场景, 涉及到不同层次和类型的统计操作. 生成大数据考察了处理大规模数据的能力, 回归分析关注了建模和预测的技术, 而高维数据主成分分析则挑战了对高维数据的降维和特征提取.

为了对比不同编程语言在这些任务上的表现, 我们选择了 R 语言、Python 和 Matlab. 这三种语言在数据科学和统计分析领域具有广泛的应用, 并代表了不同的编程范式和生态系统. 通过对它们在各个任务上所花费的时间进行比较, 我们将能够评估它们在不同统计主题下的性能表现.

此外, 我们在比较过程中将适当地采用切换底层库实现运算的加速方法. 进而, 有助于深入地了解每种语言的性能瓶颈和优化空间, 以更全面地评估它们在实际任务中的可行性.

最终的实验结果表明: Matlab 在底层逻辑库方面呈现出的相对明显的优势. 特别是在矩阵运算和并行计算方面, Matlab 展现出卓越的性能, 使得其在处理大规模数据时表现出色, 运算更加高效.

第二部分　BLAS

BLAS(Basic Linear Algebra Subprograms)是一组用于执行基本线性代数运算的软件库. BLAS 不仅包括矩阵乘法和向量运算, 还涵盖了一系列其他基本的线性代数操作, 如矩阵-矩阵加法, 矩阵-矩阵乘法, 向量内积等. 这些操作的高效实现在科学计算, 工程应用等领域起到了至关重要的作用.

BLAS 的起源, 可以追溯到 1970 年代. 当时科学计算社区对于高性能线性代数运算的需求日益增加. 为了满足这一需求, BLAS 标准应运而生, 它定义了一套常用的线性代数子程序接口, 使得科学家和工程师能够以一致的方式调用这些功能.

BLAS 的发展历程中, 有几个关键的实现库, 其中包括 Maltba(Mathematical Software for Transcendental Bundle Adjustment)以及两个主要的 BLAS 实现库, 分别是 Intel Math Kernel Library(MKL)和 OpenBLAS.

BLAS 库通常会针对特定硬件进行优化, 以获得最佳性能. 除了以上提及的两个主要实现库之外, 还有一些其他针对不同硬件架构的 BLAS 实现, 如 cuBLAS(CUDA Basic Linear Algebra Subprograms)用于 NVIDIA GPU 加速.

BLAS 的演进过程中, MKL 成为一个引人瞩目的商业库. MKL 由英特尔公司开发, 是一款高度优化的数学库, 专为英特尔架构的处理器而设计. MKL 不仅实现了 BLAS 标准, 还提供了针对英特尔处理器的优化, 使得线性代数运算能够在这些处理器上获得最佳性能. 这使得科学家和工程师在进行大规模计算时能够更有效地利用硬件资源. BLAS 的最新标准是 BLAS3, 包括了一些针对高性能计算的新操作和规范.

此外, 我们需要介绍 OpenBLAS: 一款由中国开发者共同努力创建的开源 BLAS 库. 它致力于提供高性能, 跨平台的线性代数运算支持. OpenBLAS 的发展反映了全球开源社区在科学计算领域的贡献, 为用户提供了在不同硬件平台上进行高效线性代数计算的选择. 在 Maltba 中, BLAS 的应用体现在其对于大规模捆绑调整问题的解决上. Maltba 利用了 BLAS 库的高性能特性, 通过优化的线性代数运算实现对于复杂问题的求解, 从而在计算机视觉和三维重建等领域取得了显著的成果.

BLAS 的应用也十分广泛. 作为科学计算领域的基础工具, 包括数值模拟, 信号处理, 数据分析等. 在不同实现库的支持下, 为研究人员和工程师提供了强大的数学计算能力. MKL 和 OpenBLAS 作为两个代表性的实现, 分别展现了商业和开源领域在这一领域的贡献, 推动了科学计算的发展. 深度学习框架(如 TensorFlow, PyTorch)也广泛使用了 BLAS 库, 尤其是在进行矩阵计算和神经网络训练时. 这进一步突显了 BLAS 在现代计算领域的重要性.

第三部分　如何更新 R 自带的 BLAS

在当今信息时代

第四部分　基于更新之后的 OpenBLAS 的比较

在当今信息时代

第五部分　更加详细的比较

在本部分当中, 我们进行了 R-BLAS, R-OpenBLAS, Python 和 Matlab 在 15 个不同的统计相关任务上的效率对比. 这一系列任务涵盖了广泛的统计学相关问题, 包括但不限于矩阵分解, 高维数据分析等. 通过详细的性能测量, 我们能够深入了解每个平台在处理这些任务时的表现.

这一对比不仅有助于科研人员和数据科学家选择最适合其工作需求的工具, 也为开发者提供了有关优化和性能改进的重要见解. 通过深入研究每个工具在不同任务上的表现, 我们可以更好地理解它们的优势和局限性, 从而为统计计算领域的实际应用提供更可靠的指导.

表格: 任务清单

任务编号	任务内容
1	创建高维矩阵, 进行简单转置运算或矩阵乘法运算.
2	生成高维矩阵, 并进行高次矩阵乘法运算.
3	生成 7 百万个随机值, 并对随机值进行排序.
4	生成 2800 维向量做交叉乘积.
5	基于 3000 维矩阵的回归分析.
6	对二百万个随机值做快速 Fourier 变换.
7	对于高维矩阵取特征值.
8	对于高维矩阵取行列式.
9	对于高维矩阵进行 Cholesky 分解.
10	对于高维矩阵进行求逆运算.
11	求解 Fibonacci 递推数列.
12	创建高维 Hilbert 矩阵.
13	计算多对数组的最大公约数.
14	创建高维 Toeplitz 矩阵.
15	对高维矩阵做主成分分析.

在实验结果(见附表)当中, Matlab 以自身高效的运作速度, 在多项任务运行对比当中, 拔得头筹. 同时, 我们不难发现, R-OpenBLAS 相较于 R-BLAS, 运行效率上得到了显著的提升, 更新加速库的方案卓有成效!

附表.

表格: 基于不同软件进行复杂运算的效率对比

任务编号	R-BLAS	R-OpenBLAS	Python	Matlab
1	0.115	0.125	0.117	0.079
2	0.447	0.447	0.750	0.269
3	0.560	0.552	1.427	0.310
4	0.015	0.019	0.036	0.028
5	0.007	0.003	0.002	0.001
6	0.120	0.111	0.102	0.035
7	0.744	0.355	0.406	0.193
8	1.506	0.266	0.512	0.181
9	13.152	0.581	1.073	0.426
10	1.782	0.158	2.475	1.468
11	0.210	0.218	3.922	0.020
12	0.080	0.079	0.025	0.014
13	1.197	1.189	1.316	0.151
14	0.025	0.026	0.000	0.003
15	0.002	0.001	0.001	0.014
总和	**19.962**	**4.130**	**12.164**	**3.192**

注. 在上述表格中, 蓝色字代表在同一次试验中, 运行程序最为缓慢的情形; 红色字代表在同一次试验中, 运行程序最为迅速的情形. 试验具有随机性, 不代表总体性能比较的必然性.

致　谢

本项目由上海交通大学数学科学学院 2023 级应用统计专业组织编写. 在本项目编写的过程当中:

感谢王成老师对于本项目的倾力指导!

感谢刘若禺同学对于本项目的合作与贡献!

感谢李昌骏同学对于本项目提供的诸多宝贵建议!

项目仍然存在诸多提升空间, 殷切读者们对本项目工作提供宝贵的意见. 在此, 编者表达衷心的感谢!

编 写 组

2024 年 1 月

参 考 文 献 或 链 接

[1] https://csantill.github.io/RPerformanceWBLAS.

[2]