arXiv:2602.04348v1 [math.NA] 4 Feb 2026

混合精度矩阵计算中的不精确性平衡

Erin Claire Carson*

摘要。 在新兴的高性能架构中，对多种精度和数值格式算术运算的支持正日益普及。从计算科学家的角度来看，我们的目标是确定如何以及在何处能够在代码中安全地利用混合精度计算来提升性能。在计算科学中，低精度的一种常见且自然的应用情形是：已经存在其他显著的“非精确性”来源，例如离散化误差、测量误差或算法近似误差。在这种情况下，分析这些不同非精确性来源之间的相互作用，有助于了解如何选择各种计算所用的精度，以便“平衡”误差；这样或许能够在不明显降低精度的情况下提升性能。我们介绍了这一方法的几个近期实例，展示了在数值线性代数和矩阵计算中应用混合精度的潜力。

1 引言

可用计算能力的规模在不断增长。截至2022年，我们已进入“百亿亿次时代”，这意味着我们拥有能够每秒执行 10^{18} （十亿亿次）双精度浮点运算的计算机。这种计算能力带来了巨大的科学模拟与发现机遇，但我们仍面临一项重大挑战：我们需要为实际应用设计算法和软件，使其能够充分利用强大的底层硬件。

本文关注基础数值线性代数和矩阵计算，它们构成了许多关键大规模应用的核心，包括计算流体力学、气候与天气建模、图像分析、人工智能等。关键在于，在所有这些应用中，我们执行的矩阵计算必然是不精确的 矩阵计算。

在计算科学过程中，会产生许多不同来源的非精确性。通常，我们从一个待研究的现象出发，并写下描述该现象的数学模型。当然，模型无法捕捉全部现实，因此会产生一些建模误差。为了实际求解模型方程并进行模拟，我们会产生离散化误差 和/或线性化误差。最终，我们会得到一个线性代数问题，

*查理大学数学与物理学院，carson@karlin.mff.cuni.cz。作者获得欧盟（欧洲研究委员会项目 inEXASCALE, 101075632）的资助。文中所表达的观点和意见仅代表作者本人，并不一定反映欧盟或欧洲研究委员会的观点和意见。欧盟和资助机构均不对这些观点和意见承担责任。作者还感谢查理大学研究中心项目 No. UNCE/24/SCI/005 提供的支持。

并且必须选择一种算法来求解。这会涉及我们所称的算法近似误差。例如，我们可能会对算子进行近似，或使用随机化使问题在计算上可行；也可能根据应用需求，为迭代方法选择某个停止准则。最后，当我们在真实硬件上运行该算法时，由于使用有限精度，不可避免地会产生舍入误差 。

这些误差密切相关。例如，要为迭代方法确定合适的停止准则，就必须了解其在有限精度下的局限性；而采用的具体离散化方式会导致不同的矩阵性质，进而改变迭代求解器的收敛行为。遗憾的是，这些不同的误差来源几乎总是被孤立地研究。此外，舍入误差也常常被完全忽略。

过去，在某些特定应用中，不考虑舍入误差或许情有可原，因为当时所有计算都以统一的双精度（64 位）运行。然而，如今的高性能计算硬件支持多种位宽格式，忽略舍入误差及其与其他非精确性来源的相互作用，不仅可能带来风险，也会错失利用当今硬件能力提升科学计算代码性能的机会。

2 浮点数与混合精度硬件

在我们所熟知的计算机硬件中，可用于存储数值并进行计算的位数是有限的。业界标准是 IEEE 浮点数。一个二进制浮点数可以表示为

$$(-1)^{\text{sign}} \times 2^{(\text{exponent-offset})} \times 1.\text{significand}.$$

其中有一位符号位、若干指数位，以及若干有效数位（也称为尾数或小数部分）；具体位数取决于所用格式（该格式还定义了常数偏移量）。

分配给指数的位数决定可表示数的范围，而分配给有效数的位数则会影响单位舍入误差 u，它决定了数字的存储精度。例如，对于 IEEE 754 双精度（64 位），指数位有 11 位，偏移量为 1023；显式存储的有效数位有 52 位（另有隐含的前导 1）。这意味着，最大的可表示数是 $2^{(2^{\#\text{exponent bits}}-1)-1023} = 2^{1024} \approx 2 \cdot 10^{308}$，最小的（正规）可表示数是 $2^{1-1023} = 2^{-1022} \approx 2 \cdot 10^{-308}$，单位舍入误差为 $2^{-\#\text{significand bits}} = 2^{-53} \approx 10^{-16}$。

每当我们对两个浮点数进行计算时，结果会被舍入为一个浮点数，由此产生的相对误差以单位舍入误差为界，即对于浮点数 x 和 y，

$$\text{fl}(x \text{ op } y) = (x \text{ op } y)(1+\delta), \quad |\delta| \le u, \quad \text{op} = +, -, *, /,$$

其中 fl() 表示在浮点运算中执行计算所得的结果。

在现代高性能计算机硬件上，我们可用的精度远不止双精度。例如，在 NVIDIA H100 GPU（例如，JUPITER 百亿亿次计算机所采用的 GPU）上，我们有多种不同的格式，其中并非所有格式都遵循 IEEE 754

表1： NVIDIA H100 GPU 上的浮点数格式 [22]。

	大小（位）	范围	u	Tflops 性能（NVIDIA H100 TC）
fp64	64	$10^{\pm 308}$	$1 \cdot 10^{-16}$	67
fp32	32	$10^{\pm 38}$	$6 \cdot 10^{-8}$	989
tf32	19	$10^{\pm 38}$	$5 \cdot 10^{-4}$	989
fp16	16	$10^{\pm 5}$	$5 \cdot 10^{-4}$	1979
bf16	16	$10^{\pm 38}$	$4 \cdot 10^{-3}$	1979
fp8-e5m2	8	$10^{\pm 5}$	$1 \cdot 10^{-1}$	3958
fp8-e4m3	8	$10^{\pm 2}$	$6 \cdot 10^{-2}$	3958

标准。这些格式及其属性列于表1。从表中可以看出，位数越少，指数和有效数位段可用的位数通常也越少，因此可表示数的范围通常更小，单位舍入误差也更大。不过，从最后一列可以看出，采用较低精度所带来的潜在性能提升非常显著。

低精度的性能优势已通过相对较新加入的 HPL-MxP 基准测试 [19], ——TOP500 排名中 HPL 基准测试的补充 [24]——得到凸显。截至撰写之时，当前世界上最快的计算机是劳伦斯利弗莫尔国家实验室的 El Capitan，它在 HPL 基准测试上达到 1.8 艾萨浮点运算/秒；该基准测试通过带部分选主元的高斯消元，以双精度精度求解稠密线性方程组。HPL-MxP 基准测试同样以双精度精度求解稠密线性方程组，但允许使用混合精度 迭代改进来实现这一点。在该基准测试上，El Capitan 达到 16.7 有效艾萨浮点运算/秒，展示了利用现有低精度硬件所带来的强大能力。

事实上，这是一种日益增长的趋势。TOP500 排名中配备混合精度加速器的系统数量正随时间不断增加；此外，在顶尖机器上，绝大部分浮点性能来自加速器；参见例如 [9]。若要有效利用这些计算机，我们就必须重新设计算法，使其能够利用图形处理器及其他专用加速器上可用的混合精度硬件。

3 低精度的挑战

显然，若想在当今的机器上获得接近峰值性能的表现，就必须使用低精度，但这并不意味着我们可以不加斟酌地这样做。

数值范围缩小。 首先，如表1所示，较低精度所能表示的数值范围更小。在浮点误差分析中，通常假设计算过程中不会发生上溢或下溢。然而，使用低精度时，这一假设未必成立。上溢通常会导致计算彻底失败；在某些情况下，下溢可能无伤大雅，但在另一些情况下，它会导致我们丧失重要的数值性质，例如非奇异性或正定性。

低精度的数值范围缩小会在许多方面带来挑战。从软件工程的角度来看，必须谨慎实现代码，以捕获并解决这些错误；例如，参见

[23]。从分析和算法开发的角度来看，我们希望识别问题和精度中可能导致下溢或上溢的特性，并开发相应的缓解技术。一个例子是 [18] 中开发的复杂缩放与移位方法，可将矩阵“挤入”较低精度。

无效误差界。 我们还必须考虑，在有限精度下，现有的关于算法稳定性和精度的分析是否仍然适用。在许多情况下，某个算法的后向误差或前向误差界通常只有在满足某些假设时才成立，例如 $nu < 1$，其中 n 是问题维数。使用较低精度意味着单位舍入误差 u 较大，这也意味着误差界仅对维数小得多的问题成立。例如，在半精度下， $u \approx 10^{-4}$，对于维数为 10^4 或更大的问题，我们无法保证稳定性（在百亿亿次级计算领域，这仍是相对较小的问题）。

由此引出一个问题：这究竟是算法本身的实际局限，还是算术系统的局限？抑或是标准误差分析采用最坏情况的方式，导致这些条件过于严格？例如，研究表明，采用概率方法，可以得到这样的误差界：其中 n 被替换为$\sqrt{n}$（在某些特殊情况下甚至可以替换为 1） [16]。Higham 的博客文章 [15]介绍了更多详情，并讨论了其他技术。

降低精度。 每当我们以较低精度存储一个数时，精度就会降低；每当我们以较低精度执行一次浮点运算时，精度也会降低。正如引言中所述，在科学计算的许多场景中，人们假设模型误差、离散化误差等远大于舍入误差，因此在许多分析中都会忽略舍入误差。当我们的舍入误差达到 10^{-4} 而非 10^{-16} 的量级时，这一假设可能就不再成立。因此，现在比以往任何时候都更有必要仔细分析这些误差来源并，最终权衡 各种误差。我们将在第5节给出三个相关示例。

4 混合精度迭代改进简史（非常简略）

在介绍如何权衡舍入误差与其他误差的示例之前，我们先简要（非常简略地）概述一些关于混合精度迭代改进的研究。若要了解更完整的历史及更多参考文献，请参阅 Vieubl´e [26] 的博士论文第3章。

算法1所示的迭代改进是一种用于迭代改进线性方程组 $Ax = b$ 解的算法。第1行中的初始近似解通常通过 LU分解 得到，计算出的 LU 因子会被重复用于求解第4行中的校正项 d_i 。

算法1 迭代改进

输入： 非奇异 $A \in \mathbb{R}^{n\times n}$, $b \in \mathbb{R}^n$，整数 maxit
输出： $Ax = b$的近似解 x_{i+1}

1: 求解 $Ax_0 = b$.
2: **对于** $i = 0$:maxit **执行**
3: $r_i = b - Ax_i$
4: 求解 $Ad_i = r_i$。
5: $x_{i+1} = x_i + d_i$

早在1948年，Wilkinson及其同事就已采用这种方法，在自动计算引擎 [27]，上求解线性方程组；Wilkinson还在其1963年出版的著作 [28] 中写下了该算法的有限精度（定点）分析。Wilkinson采用的传统方法使用两种精度：一种工作精度 u ，用于存储和所有计算，但第3行中的残差计算除外；该计算使用精度 u^2 （即工作精度的两倍；本文此处及下文均以“精度 u”作为“单位舍入误差为 u 的精度”的简称）。这种混合精度的用法相当自然，因为当时的硬件能够以精度 u^2 精确计算两个精度为 u 的数的标量积，且不产生额外开销。Wilkinson 的分析表明，如果 $3nu\kappa_\infty(A) < 1$，其中 $\kappa_\infty(A)$ 是 A 的无穷范数条件数，那么极限相对前向误差和后向误差，即 $\|x_i - x\|_\infty/\|x\|_\infty$ 和 $\|b - Ax_i\|_\infty/(\|A\|_\infty\|x_i\|_\infty\|b\|_\infty)$，其阶为 $O(u)$。

在 [5], 中，研究表明，只需更改第 4 行中的求解器，就能求解条件数更差的系统。具体而言，如果改用预条件 GMRES，其中预条件子由计算得到的 LU 因子构成，那么收敛条件中现在包含的是 $\kappa_\infty(U^{-1}L^{-1}A)$，而不是 $\kappa_\infty(A)$；前者可能小得多，即使 LU 因子是通过不精确计算得到的也是如此。这种方法称为基于 GMRES 的迭代改进，或简称 GMRES-IR。

受采用半精度格式的新兴硬件启发， [6]将 Wilkin- son 的方法与低精度分解变体相结合（例如参见 [20]），其中 LU分解（算法中开销最大的部分）以工作精度一半的精度计算，其余计算则以精度 u进行。由此得到了适用于一般三精度迭代改进变体的界限，该变体使用精度u_f 进行分解，使用 u_r 进行残差计算，并使用 u 进行其他计算，其中 $u_r \le u \le u_f$。分析中还包含“有效求解精度”，从而允许第 4 行使用一般求解器。

一般而言，迭代改进是适合采用混合精度的一种情形，也就是说，当算法本身包含某种“自我校正”机制，或某种“内外层”求解方案时。这在数值算法中十分常见。

5 平衡非精确性的来源

另一种适合在计算的某些环节采用低精度的情形是，算法中还存在其他显著的误差来源，例如使用低秩近似，或对计算域进行粗略近似。同样，尽管现有对这类误差的大多数分析通常忽略有限精度误差，但若要安全且有效地利用低精度计算，我们就必须仔细考察将这些类型的分析结合起来时会发生什么，以确定这些误差如何相互作用。我们的最终目标是平衡这些误差；为此，我们可以通过分析确定有限精度误差相对于其他近似误差可以有多大，才能确保它不会占主导地位。

下面将介绍三个有关这种平衡的示例，分别涉及稀疏化矩阵、随机化算法和层次矩阵近似。所有数值实验均使用 MATLAB 2025a 完成。

5.1 示例 1：混合精度稀疏近似逆预条件器

稀疏近似逆是 Krylov 子空间方法中常用的代数预条件器。全面综述请参见 [2]。目标是构造一个稀疏矩阵 M ，使得

它是 A^{-1} 的近似。在 Grote 和 Huckle [13], 提出的经典方法中（如算法2所示）， M 的每一列都可以独立构造。尽管这在理论上高度可并行，但对于大规模问题，构造过程与内存需求仍可能代价高昂；例如参见 [14]。因此，我们旨在尽可能使用低精度来构造该稀疏近似逆，以改善性能与内存需求。

算法2 稀疏近似逆构造**输入：** 非奇异 $A \in \mathbb{R}^{n\times n}$，初始稀疏结构 $\mathcal{J}$，容差 τ**输出：** 稀疏近似逆 $M \approx A^{-1}$1: **对**每一列 k **执行**2: 对由 $\mathcal{J}$定义的 A 子矩阵计算 QR分解。3: 使用 QR 求解 $\min_{m_k} \|e_k - Am_k\|_2$，其中 e_k 是单位矩阵的第 k 列。4: **若** $\|r_k\|_2 = \|e_k - Am_k\|_2 \le \tau$ **则中断；** 5: **否则** 向 $\mathcal{J}$ 中添加选定的非零元，从第 2 行重复。

在 [7], 中证明，我们可以在某种精度 u_s 下执行整个计算，并且仍能得到满足 $\|\hat{r}_k\|_2 \le \tau$ 的解（其中 $\hat{\cdot}$ 表示计算得到的量），如果

$$n^3 u_s \||e_k| + |A||\hat{m}_k|\|_2 \le \tau.$$

换句话说，如果我们希望满足停止准则，那么相对于 u_s，问题的病态程度不能太高，以免仅在计算残差时就产生大于 τ 的误差。

我们可以将其转化为一个较宽松、但或许更实用的先验界，并指出，只要满足以下条件，就可以在精度 u_s 下构造稀疏近似逆：

$$u_s \mathrm{cond}_2(A) \lesssim \tau,$$

其中 $\mathrm{cond}_2(A) = \||A^{-1}||A|\|_2$.[1] 这说明，我们必须如何平衡来自稀疏近似逆构造中停止准则的误差，以及算法产生的舍入误差。对于给定的矩阵 A 和期望值 τ，我们可以取u_s的最大值受到限制。

[7] 结果还表明，在满足某些约束的情况下，可以将该稀疏近似逆用于 GMRES 型迭代改进（取代通常的 LU分解）；此处，我们还需要约束迭代改进中的工作精度（算法1）相对于 τ的选取方式。

图1给出了一个在基于 GMRES 的 IR 中使用低精度稀疏近似逆的示例。这里我们使用 SuiteSparse 中的矩阵 steam1 [10]。我们比较了采用 LU 预条件（左图）和稀疏近似逆预处理（右图）的基于 GMRES 的 IR。预条件子（包括 LU 和稀疏近似逆）均以单精度计算，而在迭代改进过程中我们使用 $u = u_r =$ 双精度。所有情况下，GMRES收敛容差均设为 10^{-6}。对于稀疏近似逆，我们设置 $\tau = 0.1$。图1中的曲线展示了前向误差（'ferr'）、范数后向误差（'nbe'）和分量后向误差（'cbe'）；标记上方的数字表示每次迭代改进步骤中执行的 GMRES 迭代次数（即算法1第4行求解过程中的迭代次数）。在 LU 和稀疏近似逆的图中，我们

[1]这里我们用 $\lesssim$ 表示这更像是一种启发式方法，而非严格界，因为我们省去了量纲常数（反正它们在实际应用中通常会导致过高估计）。

列出了预条件子的规模，即非零元数量。由此可见，低精度稀疏近似逆是基于LU的预条件子的可行替代方案；尽管其收敛所需的 GMRES 迭代总次数略多，但应用和存储这类预条件子的成本要低得多。更多细节和示例请参见 [7]。

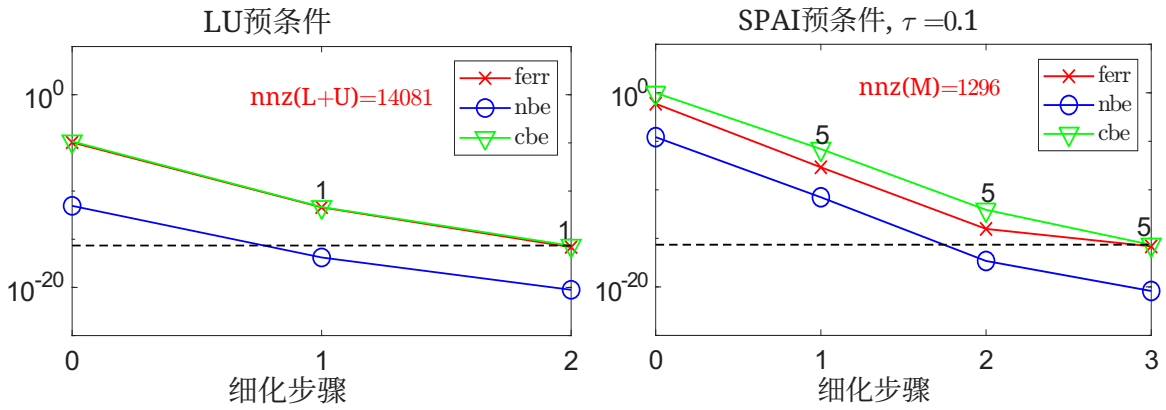


图1： 采用 LU预条件（左）和 SPAI预条件（右）的 GMRES-IR 在 SuiteSparse 中矩阵 stea m1 上的比较 [10]。标记上方的数字表示每个迭代改进步骤中执行的 GMRES 迭代次数；所有情况下，GMRES收敛容差均设为 10^{-6}。对于左上图，计算 LU 因子之前应用了 AMD排序。所有情况下，我们均使用 GMRES-IR，其中 u_f = 单精度、 u = 双精度、 u_r = 双精度。

5.2 示例2：混合精度随机化 Nystr¨om 近似

下一个示例涉及通过随机化 Nystr¨om 方法计算一个 k 秩近似，该矩阵 $n \times n$ 是对称半正定矩阵 A 。随机化 Nystr¨om 近似的形式为 $A_N = (A\Omega)(\Omega^T A\Omega)^{\dagger}(A\Omega)^T$，其中 Ω 是一个 $n \times k$ 采样矩阵，而 $\dagger$ 表示 Moore-Penrose伪逆。Nystr¨om 近似应用广泛，例如用于近似核矩阵以及构造谱有限记忆预条件器。在某些应用中，矩阵规模大到难以处理；在另一些应用中，其元素只能访问一次。这促使人们提出 Nystr¨om 方法的单遍变体 [25], ，如算法3所述。

算法3 单次遍历 Nystr¨om 近似 [25]

输入： 对称半正定矩阵 $A \in \mathbb{R}^{n\times n}$，目标秩 k**输出：** $U \in \mathbb{R}^{n\times k}$，其各列给出近似特征向量；$\Theta \in \mathbb{R}^{k\times k}$，其对角元素给出近似特征值1: $G = \text{randn}(n,\ k)$, $[\Omega\ \sim] = \text{qr}(G,\ 0)$2: $Y = A\Omega$3: 计算移位量 ν，并计算 $Y_\nu = Y + \nu\Omega$、 $B = \Omega^T Y_\nu$4: $C = \text{chol}(B + B^T)/2$，求解 $F = Y_\nu/C$5: $[U, \Sigma, \sim] = \text{svd}(F,\ 0)$, $\Theta = \max(0,\ \Sigma^2 - \nu I)$

矩阵乘　　第 2 行中的矩阵乘积在这一情形下占据了绝大部分计算成本。我们的　　r

目标因此是：除这一行以外，所有计算均采用工作精度 u ，而这一行则采用精度 $u_p \geq u$。注意，由于只在此处访问 A ，因此 A 也应以精度 u_p 存储。令 A_N 表示以混合精度计算得到的 Nyström 近似，则可写为 $\|A - A_N\|_2 \leq \|A - A_N\|_2 + \|A_N - A_N\|_2$；换言之，我们将总误差界定为精确近似误差与有限精度误差之和。第一项的界此前已有推导；例如，参见 [12], [11]。在 [4], 中，我们证明了有限精度误差的第一个上界。结果基本表明，只要选择精度 u_p 和 u，使得 $\kappa_2(A_k)\tilde{\kappa}(\Omega)^2 \ll u_p^{-2}$ 且 $\kappa_2(A_k)\tilde{\kappa}(\Omega)^2 \ll u^{-1}$，那么

$$\|A_N - \hat{A}_N\|_F \lesssim \|A - A_N\|_F + k^{1/2} n u_p \kappa_2(A_k)\tilde{\kappa}(\Omega)^2 \|A\|_F,$$

其中， A_k 是 A 的最佳秩-k 近似， $\tilde{\kappa}(\Omega) = \|\Omega\|_F \|(W_1^T \Omega)^\dagger\|_2$，其中 W_1 是 A中对应于前几个 k 特征值的特征向量。在某些假设下，我们可以利用这个界推导出一种启发式方法：很可能有限精度误差小于近似误差，只要

$$u_p \leq n^{-1/2} \lambda_k / \lambda_1,$$

其中， λ_1 和 λ_k 分别是 A中最大的特征值和第 k大的特征值。这再次告诉我们如何平衡误差：近似误差越大（ λ_k 越接近 λ_1），我们就能安全使用越低的精度。在图 2 中，我们以 SuiteSparse 中的矩阵 bcsstm07 [10]为例进行说明。从左图可以看出，根据上述启发式方法，对于 $k \leq 148$, 我们可以安全地使用半精度，也始终可以安全地使用单精度。事实上，右图中的总误差似乎证实了这一点：单精度与精确情形没有差异，而半精度下直到$k > 150$附近才开始出现有限精度误差占主导的情况。更多示例、另一种（可能更严谨的）启发式方法，以及关于如何选择 Ω 和 ν的见解，请参见 [4]。

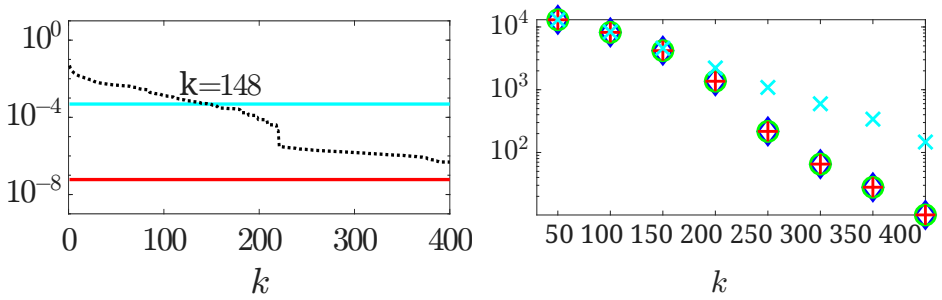


图 2： 矩阵 bcsstm07 的实验。左图显示缩放后的谱 λ_k/λ_1 （黑色虚线），以及半精度（青色）和单精度（红色）下的 $\sqrt{n u_p}$ 值。启发式方法表明，当 k 小于黑色虚线与彩色线相交处对应的值时，有限精度误差不应占主导。右图显示五次运行中平均 $\hat{}$ 总误差 $\|A - A_N\|_F$ 的弗罗贝尼乌斯范数：精确算术（蓝色菱形）、 u_p， u = 双精度（绿色圆圈）、 u_p = 单精度与 u = 双精度（红色 +s），以及 u_p = 半精度与 u = 双精度（青色 ×s）。

5.3 示例 3：混合精度分层矩阵近似

最后一个示例涉及层次矩阵近似，尤其是 HODLR（分层非对角低秩）矩阵。HODLR 矩阵具有固定的分层块结构；

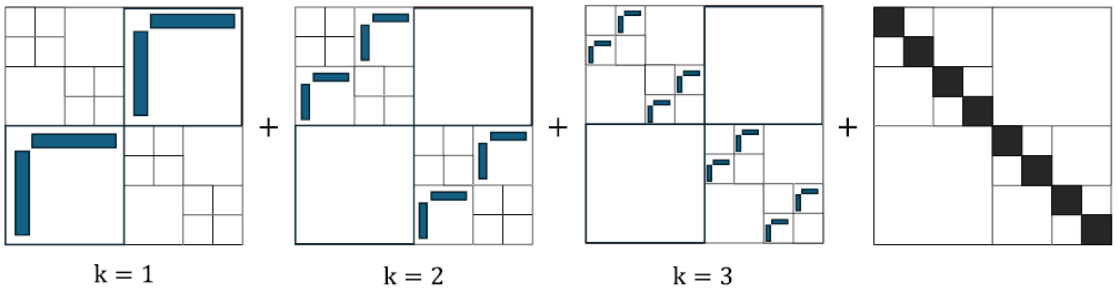


图3： 一个具有 $\ell = 3$ 层的 HODLR 矩阵示意图

给定层数 ℓ，我们将每一层的非对角块存储为秩为 p 的近似，并将最后一层的对角块存储为稠密矩阵。请参见图 3 中的示意图 $\ell = 3$ 。这类表示可降低计算成本，例如， $O(n^2)$ 计算可变为 $O(pn\log\ n)$。

其思路是，既然我们已经在近似非对角块，那么很可能也可以用较低精度存储它们，而不会损失大量额外信息。文献 [3] 对此进行了分析。

设 ε 为 HODLR 矩阵 H 的非对角块近似中的最大相对误差（例如，由使用截断奇异值分解产生）。设 H 为 H 的混合精度表示，其中第 k 层的非对角块以精度 u_k 存储，而最后一层的对角块以工作精度 $u < u_k$ 存储。那么，全局近似误差的上界可写为

$$\frac{\|H - \hat{H}\|_F}{\|H\|_F} \lesssim 2\sqrt{2}\left(\sum_{k=1}^{\ell} 2^k \xi_k^2 u_k^2\right)^{1/2} + \varepsilon,$$

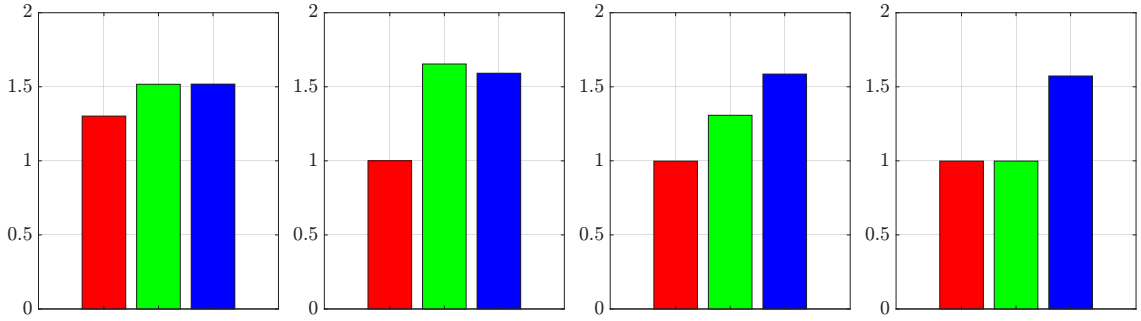
其中 $\xi_k = \max_{|i-j|=1} \|H_{ij}^{(k)}\|_F / \|H\|_F$，而 $H_{ij}^{(k)}$、$|i-j| = 1$ 表示第 k 层中的任意一个非对角块。

这再次告诉我们如何平衡误差：如果我们选择

$$u_k \le \varepsilon/(2^{k/2}\xi_k),$$

那么上述界的右侧可简化为（$2\sqrt{2\ell+1}$）ε；换句话说，我们看不出以较低精度存储非对角块所带来的影响。 [3], 基于这一思路，本文提出了一种用于 HODLR矩阵构造的自适应精度方法。图 4 展示了针对 SuiteSparse [10] 中若干问题，采用不同 ε 时所获得的存储节省示例。

在 [3] 中，我们还确定了后续使用混合精度 HODLR 矩阵进行计算时应如何设置工作精度 u ，以确保计算的后向误差不超过矩阵非精确表示所产生的误差。这些定理表明，对于以混合精度格式存储的 HODLR 矩阵，无论是计算矩阵-向量乘积还是进行 LU分解，只要满足 $u \le \varepsilon/n$，这些计算的后向误差就会受矩阵表示误差主导。同样，这为我们提供了一个理论依据，使我们能够安全地平衡这些误差。详情和示例请参见 [3]。



(a) nos7 (b) psmigr 1_ (c) saylr3 (d) 1138 总线

图 4： 自适应精度 HODLR 矩阵相对于统一（双精度）HODLR 矩阵的存储节省。深度为 $\ell = 8$；红色柱表示 $\varepsilon = 10^{-7}$，绿色柱表示$\varepsilon = 10^{-4}$，蓝色柱表示 $\varepsilon = 10^{-1}$。

6 其他示例与展望

还有许多其他平衡有限精度误差与其他误差的例子，例如离散化和采样误差 [21], ，以及硬件故障 [8], ，但篇幅所限，无法在此一一列举。对于对混合精度数值算法感兴趣的读者，我们推荐阅读综述 [1] 和 [17]。

展望未来，我们显然正越来越多地迈向大规模系统极端异构的时代。这意味着，采取整体性方法进行算法设计，并真正考虑整个计算科学流程（包括运行代码所依托的硬件），比以往任何时候都更加重要。

目前，许多工作正致力于开发新的非 IEEE 数字格式和算术系统、开发近似硬件，以及推动量子计算、神经形态计算及其混合形式等新型计算范式的兴起。在这些新范式中，分析多种来源的误差，并确定如何以最优方式平衡误差，必将长期带来有趣且富有挑战性的问题。

最后，我们希望读者思考自己所在的应用领域、其中涉及的误差来源，以及在何种情况下采用低精度或混合精度计算可能更为合适。

致谢。 本文谨献给 Nicholas J. Higham，以纪念这位导师与朋友。他最初激发了我对混合精度算法的兴趣。

参考文献

[1] Ahmad Abdelfattah、Hartwig Anzt、Erik G Boman、Erin Carson、Terry Cojean、Jack Dongarra、Alyson Fox、Mark Gates、Nicholas J Higham、Xiaoye S Li 等。利用混合精度算术的数值线性代数方法综述。国际高性能计算应用期刊，35(4):344–369，2021年。[2] Michele Benzi。大型线性系统的预条件技术：综述。计算物理学杂志，182(2):418–477，2002年。

[3] Erin Carson、Xinye Chen 和 Xiaobo Liu。混合精度 HODLR 矩阵。*SIAM* 科学计算杂志，47(3):A1408–A1435，2025年。[4] Erin Carson 和 Ieva Dauˇzickaitˇe。混合精度下的单遍 Nyström 近似。*SIAM* 矩阵分析与应用杂志，45(3):1361–1391，2024年。[5] Erin Carson 和 Nicholas J Higham。迭代改进及其在病态稀疏线性系统精确求解中的应用：新分析。*SIAM* 科学计算杂志，39(6):A2834–A2856，2017年。[6] Erin Carson 和 Nicholas J Higham。通过三种精度下的迭代改进加速线性系统求解。*SIAM* 科学计算杂志，40(2):A817–A847，2018年。[7] Erin Carson 和 Noaman Khan。结合稀疏近似逆预处理的混合精度迭代改进。*SIAM* 科学计算杂志，45(3):C131–C153，2023年。[8] Erin Claire Carson 和 Jakub Hercík。流水线 Krylov 子空间方法中静默错误的检测与纠正。数值算法，第 1–36 页，2025年。[9] Noel Chalmers、Jakub Kurzak、Damon McDougall 和 Paul Bauman。针对百亿亿次级加速架构优化高性能 Linpack。载于国际高性能计算、网络、存储与分析会议论文集，第 1–12 页，2023年。[10] Timothy A Davis 和 Yifan Hu。佛罗里达大学稀疏矩阵集合。*ACM* 数学软件汇刊，38(1):1–25，2011年。[11] Zachary Frangella、Joel A Tropp 和 Madeleine Udell。随机化 Nyström 预处理。*SIAM* 矩阵分析与应用杂志，44(2):718–752，2023年。[12] Alex Gittens 和 Michael W Mahoney。重新审视 Nyström 方法，以改进大规模机器学习。机器学习研究杂志，17(1):3977–4041，2016年。[13] Marcus J Grote 和 Thomas Huckle。利用稀疏近似逆进行并行预条件。*SIAM* 科学计算杂志，18(3):838–853，1997年。[14] Guixia He、Renjie Yin 和 Jiaquan Gao。GPU 上一种高效的稀疏近似逆预处理算法。并发与计算：实践与经验，32(7):e5598，2020年。[15]Nicholas J. Higham。我们能否在极端规模和低精度下求解线性代数问题? https://nhigham.com/2021/09/14/ can-we-solve-linear-algebra-problems-at-extreme-scale-and-low-precisions/，2021年。[16] Nicholas J Higham 和 Theo Mary。概率舍入误差分析的新方法。*SIAM* 科学计算杂志，41(5):A2815–A2835，2019年。[17] Nicholas J Higham 和 Theo Mary。数值线性代数中的混合精度算法。数值学报，31:347–414，2022年。

[18] Nicholas J Higham、Srikara Pranesh 和 Mawussi Zounon。《将矩阵压缩为半精度，并将其应用于求解线性方程组》。*SIAM* 科学计算杂志，41(4):A2536– A2551，2019年。[19] HPL-MxP混合精度基准测试。https://hpl-mxp.org，2024年。[20] J. Langou、J. Langou、P. Luszczek、J. Kurzak、A. Buttari 和 J. Dongarra。《利用32位浮点运算的性能获得64位精度（重新审视线性方程组的迭代改进）》。载于*2006*年*ACM/IEEE*超级计算会议论文集，2006年。[21] Josef Mart´ınek、Erin Carson 和 Robert Scheichl。《在多层采样方法中利用不精确计算》。*arXiv*预印本 *arXiv:2503.05533*，2025年。[22] 英伟达。NVIDIA H100张量核心GPU架构。https://resources.nvidia.com/ en-us-hopper-architecture/nvidia-h100-tensor-c，2022年。[23] Jennifer Scott 和 Miroslav T˚uma。《以半精度浮点运算开发稳健的不完全Cholesky分解》。数值算法，第1–22页，2025年。[24] TOP500：榜单。https://top500.org/，1993年。访问于2026年1月。[25] Joel A Tropp、Alp Yurtsever、Madeleine Udell 和 Volkan Cevher。《根据流式数据对半正定矩阵进行固定秩近似》。神经信息处理系统进展，30，2017年。[26] Bastien Vieubl´e。《大型稀疏线性方程组求解的混合精度迭代改进》。博士论文，图卢兹大学，法国图卢兹，2022年。[27] J. H. Wilkinson。《自动计算引擎进展报告》。技术报告MA/17/1024，科学与工业研究部数学部，国家物理实验室，泰丁顿，英国，1948年4月。127页。[28] James H. Wilkinson。《代数运算中的舍入误差》，应用科学札记第32卷。女王陛下文书局，伦敦，1963年。