基于误差建模的位宽联合优化

调研报告

**许威，教授**

**东南大学移动通信国家重点实验室**

目录

[第一章 前言 1](#_Toc150697140)

[1.1 背景 1](#_Toc150697141)

[1.2 现有方法 4](#_Toc150697142)

[1.2.1 存内计算 4](#_Toc150697143)

[1.2.2 分布式学习与联邦学习 5](#_Toc150697144)

[1.2.3 可变精度计算 6](#_Toc150697145)

[第二章 低精度浮点数四则运算方法[43] 8](#_Toc150697146)

[2.1 加减法 8](#_Toc150697147)

[2.2 乘法 11](#_Toc150697148)

[2.3 除法 14](#_Toc150697149)

[第三章 低复杂度矩阵求逆算法 18](#_Toc150697150)

[3.1 QR分解法 18](#_Toc150697151)

[3.2 LU分解法 19](#_Toc150697157)

[3.3 Sherman-Morrison求逆法 20](#_Toc150697158)

[3.4 QR,LU和SM求逆算法对比 20](#_Toc150697159)

[3.5 Neumann级数展开 21](#_Toc150697160)

[3.6 Cholesky分解 22](#_Toc150697161)

[3.7 CSM算法 23](#_Toc150697162)

[3.7.1 具体算法 23](#_Toc150697163)

[3.7.2 CSM与Neumann级数对比 25](#_Toc150697164)

[第四章 低复杂度矩阵SVD算法 26](#_Toc150697165)

[4.1 Householder变换与QR分解计算SVD 26](#_Toc150697166)

[4.2 随机奇异值分解 28](#_Toc150697167)

[4.3 Lanczos迭代SVD 30](#_Toc150697168)

[4.4 Jacobi求解SVD 32](#_Toc150697169)

[第五章 面向极低精度的可变精度计算深化设计 34](#_Toc150697170)

[5.1 可变精度计算设计概述 34](#_Toc150697171)

[5.2 下一阶段优化方向 34](#_Toc150697172)

[5.2.1 指标设计优化 34](#_Toc150697173)

[5.2.2 尺度设计优化 35](#_Toc150697174)

[5.2.3 效用函数设计优化 35](#_Toc150697175)

[5.2.4 超低精度(6-8bit)可变精度计算设计 36](#_Toc150697176)

[第六章 低精度预编码 37](#_Toc150697177)

[6.1 低精度预编码 37](#_Toc150697178)

[6.2 低复杂度预编码 38](#_Toc150697179)

[6.3 非完美CSI建模 41](#_Toc150697180)

[6.3.1 非完美CSI建模过程[50] 41](#_Toc150697181)

[6.3.2 非完美CSI情况下的资源分配(数据位数和功率的分配)[51] 43](#_Toc150697182)

[第七章 总结与研究思路 46](#_Toc150697183)

[第八章 参考文献 47](#_Toc150697184)

# 前言

## 背景

通感算一体化网络是指同时具备物理-数字空间感知、泛在智能通信与计算能力的网络。该网络内的各网元设备通过通感算软硬件资源的协同与共享，实现多维感知、协作通信、智能计算功能的深度融合、互惠增强，进而使网络具备新型闭环信息流智能交互与处理及广域智能协作的能力。

通信、感知、计算三者深度融合、互惠增强，其逻辑关系可以从三个方面进行解读：

1. 通信、计算辅助感知增强
2. 感知、计算辅助通信增强
3. 感知增强与通信增强进一步辅助计算增强

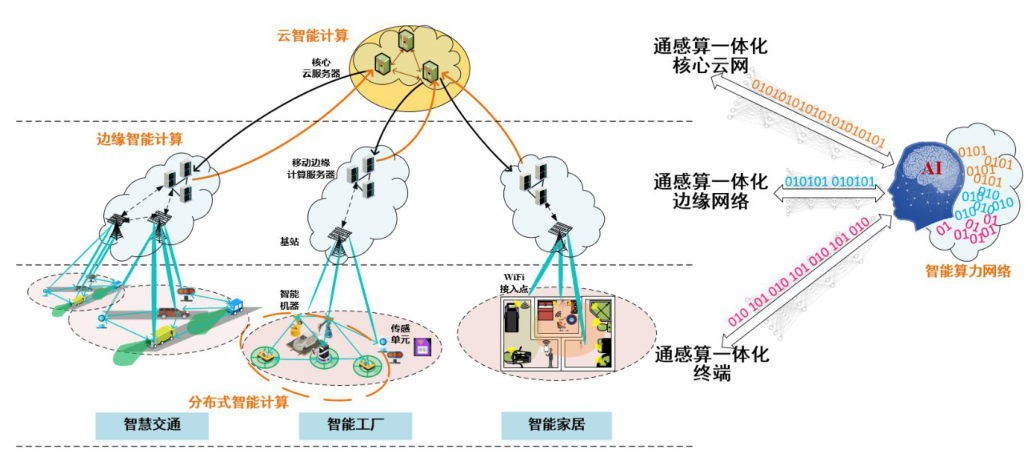


图 1-1 通感算一体化网络架构图

通感算赋能的算力网络是实现云网边端统控的新型网络架构，是云网融合体系中的一种关键技术，其通过网络控制面(包含集中式控制器、分布式路由协议等)分发服务节点的算力、存储、算法等资源信息，并结合网络信息和上层应用(如6G应用或6G网络功能单元等)需求，提供最佳的计算、存储、网络等资源的分发、关联、交易与调配，从而实现整网资源的最优化配置和使用。通感算赋能的算力网络架构如下图1-2所示：

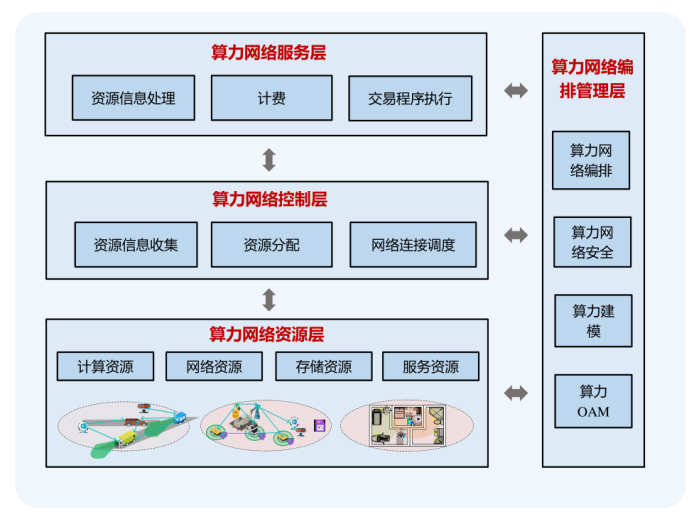


图 1-2 通感赋能的算力网络架构

移动算力网络是在6G网络架构的控制层面，引入通信与计算深度协同的功能、接口和协议，直接通过未来3GPP标准化的信令来实现通算融合甚至一体化。移动算力网络融合的算力资源可以来自网络设备内生的计算，也可以来自中心云、边缘云、终端和等，也就是说只要符合未来标准化的通算融合机制的合规算力资源，都将是移动算力网络的目标资源。移动算力网络将构建出一种开放的机制和生态，来高效的整合网络内外的各类算力资源，并基于网络架构内生的通算融合机制，为6G业务创新提供新型的基础设施。

未来大量6G应用会是计算密集型的，如AI、XR、多模态通信、数字孪生等，同时这些应用往往还有极致性能的要求。移动算力网络需要为能力受限(电量、算力等)的移动终端按需提供实时的计算服务，并在无线动态环境下高效的保障计算任务QoS(计算精度/时延/能效、计算交互时延/抖动等)。同时，移动算力网络将是6G各类内生智能、感知等计算型服务的基础平台能力。

近几年芯片发展速度减缓，预测芯片发展规律的摩尔定律接近物理极限，下一代工艺的进一步更新正在放缓，芯片的能效进步陷入了瓶颈，这使得我们需要新的计算架构来突破制程的限制。

当下的系统绝大多数都为冯诺依曼架构，这种架构有多个数据存储媒介，使得数据需要经常性地在不同媒介中移动以满足计算和存储的需要。然而这种设计直接导致了如下三方面问题。1)随着应用变得更加数据密集，内存带宽的发展速度远不及需求增长速度，来自内存的数据访问已经是一个关键瓶颈；(2)在移动和服务器系统中，能源消耗在系统中的比重与日俱增；(3)数据移动在带宽、能源和延迟方面的影响使得其比计算消耗的影响更大。这些趋势在当今数据密集型的服务器和能源紧张的移动系统中尤为明显。同时，传统的内存技术在可靠性、能源和性能方面正面临着许多挑战。

主存储器是使用动态随机存取存储器(DRAM)技术建立的，是几乎所有计算系统中的一个主要组成部分。在所有这些系统中，现代应用的数据工作集大小正在迅速增长，导致主存储器成为这些应用的一个重要瓶颈。造成主存储器瓶颈的一个主要原因是数据移动成本过高。在今天的计算机中，要对驻留在主存储器中的数据进行任何操作，存储器中的控制器必须首先通过片外总线向DRAM模块发出一系列命令。DRAM模块通过存储器通道向存储器控制器发送数据，然后将数据放在缓存或寄存器中[1]。只有在数据进入缓存后，CPU才能对数据进行操作。这个将数据从DRAM转移到CPU的过程会产生很长的延迟[2]，并消耗大量的能量。

数据移动的消耗是当代计算机系统以处理器为中心的一个基本问题。中央处理器被认为是系统中的主宰，计算只在处理器中进行。相比之下，数据存储和通信单元，包括主存储器，被视为没有智能的工人，无法进行计算。由于这种以处理器为中心的设计范式，数据在系统中的计算单元和通信/存储单元之间大量移动，以便对其进行计算。随着新兴应用越来越以数据为中心，以处理器为中心的设计范式导致了性能、能源和成本的效率很低。例如，单个计算节点内的大部分资源已经被用于处理数据移动和存储(例如，大型缓存、内存控制器、内部连接和主内存)。最近的工作表明[3]，对于广泛使用的移动工作负载，移动设备的整个系统能量的62%以上被用于处理器和内存层次之间的数据移动。

当下内存设计和内存架构的最新进展使我们有机会向存内计算(PIM)的模式转变，我们可以重新设计计算机，使其不再以处理器为中心，并避免不必要的数据移动。存内计算，也被称为近数据处理(NDP)，使我们能够使用内存本身或一些处理逻辑(如加速器、简单内核或可编程器件)来执行操作。存内计算提出至少已有四十年了，然而由于各种原因，过去的研究没有被大规模采用，其中包括将处理元件与DRAM集成上的困难，以及当时内存技术的发展速度要远快于今日以至于矛盾不够突出。由于现代内存架构的进步，例如以3D堆叠方式集成逻辑和内存，最近的各种研究探索了一系列用于多种不同目的的PIM架构。

分布式机器学习也称分布式学习(Distributed Learning)是一种机器学习方法，是指利用多个计算节点(Worker)进行机器学习或者深度学习的算法和系统，旨在提高性能、保护隐私，并可扩展至更大规模的训练数据和更大的模型。现有的分布式学习平台主要包括TensorFlow、PyTorch、MxNet等。

几十年来对机器学习中大部分计算的操作进行优化的需求一直是高性能计算领域高度活跃的研究方向，一些来自高性能计算社区的技术和库已经被机器学习社区成功采用并运用到系统中。同其他大规模运算一样有两种基本不同且互补的方式来加速工作负载：向一台机器添加更多的资源(垂直扩展，比如GPU\TPU计算核心的不断提升)，向系统添加更多的节点(水平扩展)。由于向系统添加节点的成本较低，因此机器学习工作负载的并行化对大规模实现可接受的性能至关重要。

在计算机中，实数是用有限数量的0和1来表示的，即有限精度，这意味着根据所使用的位数，有有限数量的可精确表示的数字。如图1-3所示，在64位精度中许多位无意义的比特被存储和计算，这种资源过度浪费了带宽和效能。且由于物理限制，基于传统技术的计算能力的增长正在放缓，因此为了继续提高效能，许多工作开始考虑可变精度的计算(Variable Precision Computing, VPC)。

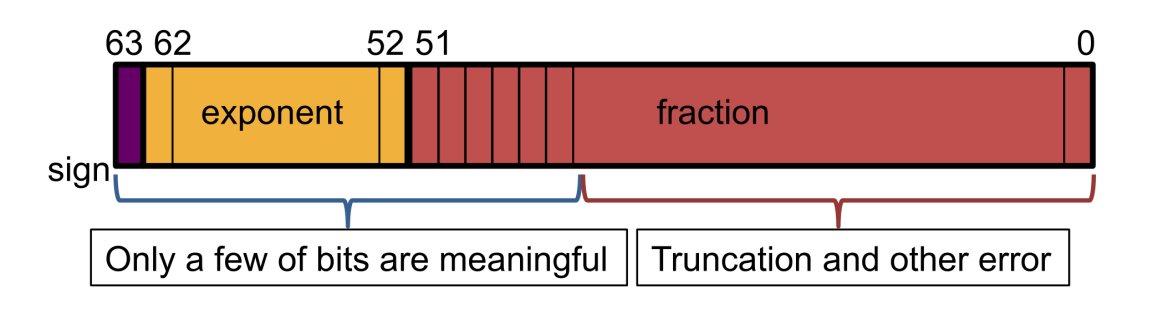


图1-3 64位精度

## 现有方法

本节将探讨在现代系统中实现降低算力，提高计算性能的一些现有方法，其中包括存内计算，分布式学习与联邦学习以及可变精度计算。

### 存内计算

现有的存内计算有两种方法。第一种方法是对内存芯片进行最小化改变[4]，以执行简单却强大的基础操作，这些操作是可以通过芯片以非常高的效率执行。我们称这种方法为直接内存处理(processing using memory, PUM)。一些属于这种方法的方案利用现有的DRAM设计[6]，巧妙而有效地使用批量操作(即对一整排DRAM单元的操作)，如批量复制、数据初始化和位操作。其他解决方案利用新的非易失性存储器技术的模拟操作原理来执行类似的批量操作或其他单独的计算，如卷积和矩阵乘法[7]。

第二种方法是通过利用传统内存控制器的计算能力或相对较新的3D堆叠内存技术的逻辑层[9]，以实现更通用的存内计算。这种更通用的方法被称为近内存处理(processing near memory, PNM)。这种方法被最近的3D堆叠存储器技术的进步所带动，该技术包括存储器层下面的逻辑处理层。为了堆叠多层存储器，三维堆叠芯片使用垂直硅通孔(TSVs)来连接各层存储器，来连接芯片的各层以及I/O。TSVs在三维堆叠层内提供的内部带宽比外部的内存通道要大得多。一些这样的三维堆叠内存架构，如混合内存立方体和高带宽内存(HBM)，包括一个逻辑层，设计者可以在其中添加一些处理逻辑(如加速器、简单的内核、可重构逻辑)，以利用这种高内部带宽。未来的芯片堆叠技术，如monolithic3D[11]，可以通过极大地提高内部带宽和内存层之间的逻辑层数量来放大这种方法的优势。

### 分布式学习与联邦学习

一般来说，机器学习问题可以分为训练阶段和预测阶段。训练阶段包括训练一个机器学习模型，通过输入大量的训练数据，并使用常用的ML算法，如进化算法(Evolutionary Algorithm，EA)、基于规则的机器学习算法(Rule-based Machine Learning algorithm)、主题模型(Topic Model，TM)、矩阵分解(Matrix Factorization)和基于随机梯度下降(Stochastic Gradient Descent，SGD)的算法等，进行模型更新。

当涉及分布式时，可以用两种不同的方法将问题划分到所有机器上，即数据或模型并行，这两种办法也可以同时应用。在数据并行方法中，系统中有多少工作节点，数据就被分区多少次，然后所有工作节点都会对不同的数据集应用相同的算法。相同的模型可用于所有工作节点，因此可以自然地产生单个一致的输出。该方法可用于在数据样本上满足独立同分布假设的每个ML算法(即大多数ML算法)。在模型并行方法中，整个数据集的精确副本由工作节点处理，工作节点操作模型的不同部分。因此，模型是所有模型部件的聚合。模型并行方法不能自动应用于每一种机器学习算法，因为模型参数通常不能被分割。一种选择是训练相同或相似模型的不同实例，并使用集成之类的方法(如Bagging、Boosting等)聚合所有训练过的模型的输出。最终的架构决策是分布式机器学习系统的拓扑结构。组成分布式系统的不同节点需要通过特定的体系结构模式进行连接，以实现丰富的功能。这是一个常见的任务。然而，模式的选择对节点可以扮演的角色、节点之间的通信程度以及整个部署的故障恢复能力都有影响。

分布式机器学习发展到现在，也产生了隐私保护的一些需求，从而与联邦学习产生了一些内容上的交叉，其中联邦学习是利用分布式资源协同训练机器学习模型的一种有效方法。联邦学习是一种分布式机器学习方法，其中多个用户协同训练一个模型，同时保持原始数据分散，而不移动到单个服务器或数据中心。在联邦学习中，原始数据或基于原始数据进行安全处理生成的数据被用作训练数据。联邦学习只允许在分布式计算资源之间传输中间数据，同时避免传输训练数据。分布式计算资源是指终端用户的移动设备或多个组织的服务器。联邦学习将代码引入数据，而不是将数据引入代码，从技术上解决了隐私、所有权和数据位置的基本问题。这样，联邦学习可以使多个用户在满足合法数据限制的同时协同训练一个模型。

分布式学习主要强调在一个可信计算集群内，发挥多个较强的计算单元协同计算的作用，通过各个计算节点的训练和集群内的通信，加速训练的过程，能够比只依赖单个较强的计算节点训练得更快。而联邦学习主要强调在复杂的网络环境中，以一种尽可能保护用户数据隐私的方式，有服务器端分发当前轮次的全局模型，基于此模型，以数据并行的方式在设备端进行分布式训练本地模型，并发送模型参数更新，在服务器端聚合，在多轮协同训练过程后能够收敛得到一个性能较好的全局模型，能够比大多数单个设备只依靠本地数据完成的本地训练提取出的模型更好。[12]

### 可变精度计算

许多工作着力于可变精度计算的探索，其中多种硬件和程序的实现使用了新的数据格式：UNUM[13]。

John L. Gustafson[16]提出了一个新的数据格式——通用数(Universal number, UNUM)，其包含所有的标准浮点格式以及定点和精确整数算法，UNUM能比浮点运算得到更精确的答案，并且在许多情况下使用更少的比特，从而节省内存、带宽和能量。UNUM能计算环境动态自动调整精度和动态范围，无需人们刻意地选取哪种精度， 比浮点数运算更容易使用也更安全，其缺点在于它需要在芯片中实现更多的门电路。

UNUM的格式可以由下图1-4所表示：

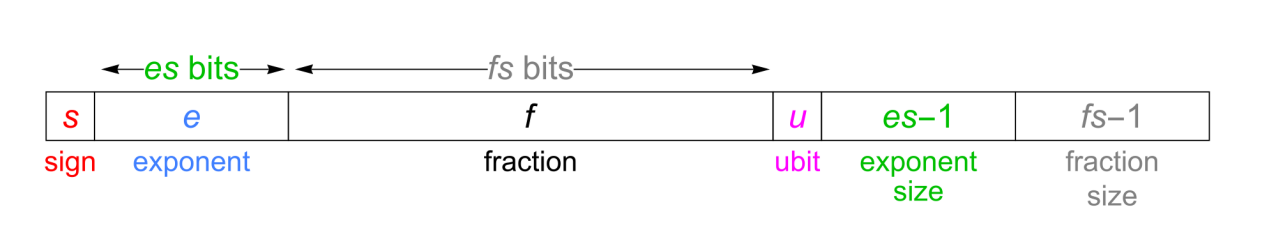


图1-4 UNUM的格式

其左边三个字段类似于IEEE的浮点数，但是有更好的关于非整数和无穷大的规则以及更好的处理溢出和舍入的方法。由于其使用不确定位，作者将其称为ubit。

ubit还可以表示一个介于最大可表示实数和无穷大之间的实数或者介于最小可表示数与零之间的情况。简而言之，UNUM通过将不确定性变得确定并存储在其数字的自我描述当中。

文献[17]在UNUM的基础上提出了一种适用于高精度及可变精度计算的创新浮点架构SMURF加速器(Scalar Multiple-precision Unum Risc-V Floating-point)，其是作为由Rocket-chip生成器生成的RISC-V处理器的协处理器实现的。协处理器使用了两种浮点格式：用于主存储器的UNUM格式和用于内部计算的gbound。在主存中，协处理器支持多个UNUM环境，最高可达16和256比特的指数和尾数。

在Lawrence Livermore国家实验室开发的一种有损的压缩算法ZFP压缩浮点数组表示的基础上，文献[18]将这种格式与分层多分辨率数据格式相结合，以提供“数据优化”表示，可以根据需要优化或粗化网格分辨率和精度，从而在可视化和数据分析目的的误差预算下最小化数据。其还对底层数据以ZFP压缩数组格式表示的计算的舍入误差累积进行严格的分析，并扩展ZFP表示，使其具有本地适应性，从而进一步增加数据减少方面的有效收益。为了减少计算量，其试图将后验误差估计的误差传输方法从截断误差扩展到舍入误差估计，并考虑将其与AMR相关的局部、基于补丁的方法相结合。

# 低精度浮点数四则运算方法[43]

## 加减法

浮点数加减运算当中，浮点数加减运算都是转换成补码进行运算的，而且所使用的浮点数都是规格化浮点数，且两浮点数的运算结果也必须是规格化数，即符合 IEEE-754标准浮点数格式。假设有两个规格化的浮点数、，， 为符号位，，为阶码，为偏移量，，为尾数其标准的表达式(2-1)所示。

(2-1)

在表达式(2-1)中令，则两浮点数和的加减运算结果有两种形式如(2-2)和(2-3)所示。

(2-2)

(2-3)

通过两浮点数X1与X2的加减运算过程，可以得出两浮点数加减运算基本流程需要以下几个步骤：

(1)指数对阶：阶码减去偏移量，得到浮点数的实际指数值，比较两个指数的大小，只有当两个浮点数指数相等才可以加减运算，因为当浮点数的指数不相等时，则浮点数的尾数的小数点位置就没有对齐。其次对阶使用的方法是“小阶向大阶看齐”的原则，这样做的目的是减少计算过程中产生的误差。通过两指数相减，求出两指数的差值，然后将指数小的浮点数的尾数向右移差值位，使两个浮点数的指数能够相同，然后才能进行下一步运算；

(2)尾数相加减：在浮点数加减运算器中，先把包含隐含位的有效尾数转换成补码，然后在进行加减运算；

(3)检查尾数：尾数相加减后，把补码结果转换成原码，判断尾数是否发生溢出，若发生溢出，尾数右移并且调整指数，若没有发生溢出，则进行下一步运算；

(4)前导 1 检查：查找加减运算结果中第一个 1 的位置，将第一个 1 移至规定的位置，并且调整指数；

(5)溢出判断：在调整指数时，可能出现指数溢出的情况，若指数发生溢出，则进行标志。

其过程可以用图 2-1 表示。

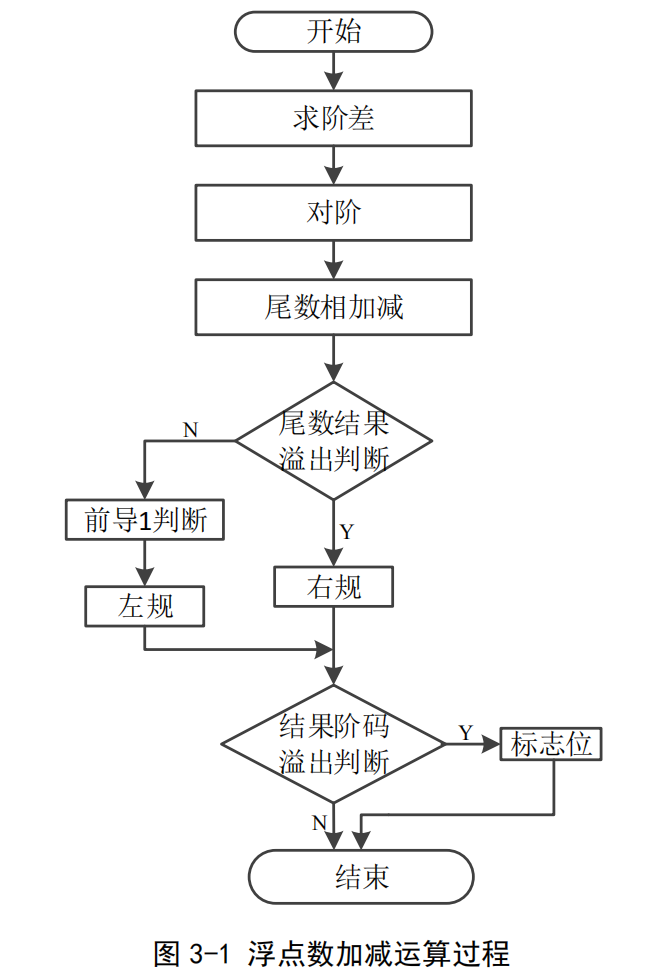


图2-1 浮点数加减法运算过程

上述就是浮点数加减法计算的基本运算步骤，在浮点数加减器运算过程中，可能会人为粗心的输入一些特殊数，因此需要对输入的浮点数做检查，此部分在结构设计过程中已经充分考虑到并且实现，其次在设计单双精度浮点加减运算器结构时，可能会根据一些具体的情况，在浮点运算器结构中会略去某个步骤或者增加一些内容，但是浮点数的加减运算步骤与上述运算步骤基本相同。

S对于单精度尾数相加，并行加法器将有 23 个全加器(full-adder)组成。n 个全加器通过级联就构成了 n 位的并行加法器，图中 C-1，C1，……，Cn-1 为进位，如图 2-2 所示。

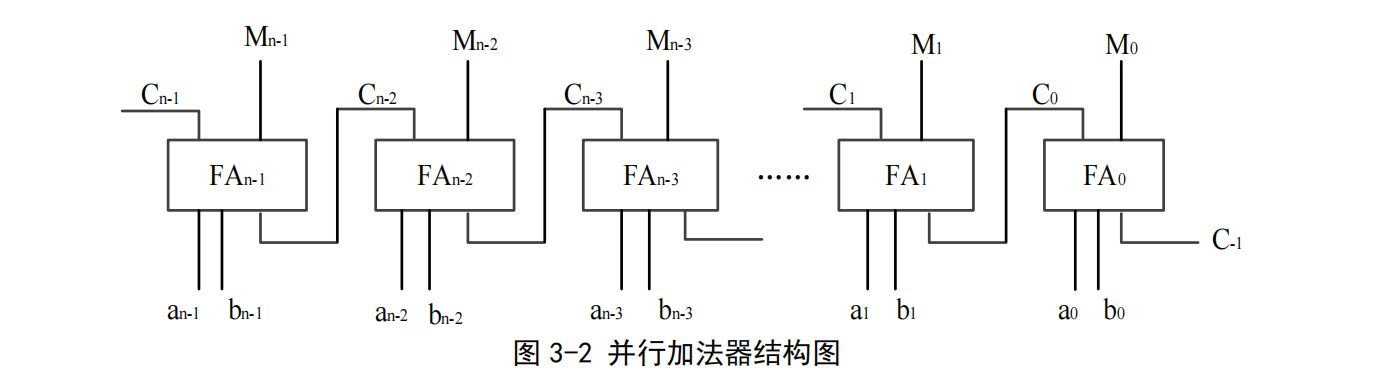


图2-2 并行加法器结构图

假设存在两个单精度浮点数尾数A和B，假设ai，bi为浮点数尾数A和B某一相同位置上的数值，将ai，bi，Ci-1相加，输出Mi和Ci，Mi和Ci的真值表如表2-1所示

表2-1 全加器运算真值表

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 输入 | | | 输出 | |
| ai | bi | Ci-1 | Mi | Ci |
| 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 1 | 1 | 0 |
| 0 | 1 | 0 | 1 | 0 |
| 0 | 1 | 1 | 0 | 1 |
| 1 | 0 | 0 | 1 | 0 |
| 1 | 0 | 1 | 0 | 1 |
| 1 | 1 | 0 | 0 | 1 |
| 1 | 1 | 1 | 1 | 1 |

根据表2-1可以得到Mi和Ci关于ai，bi，Ci-1的表达式为

(2-4)

(2-5)

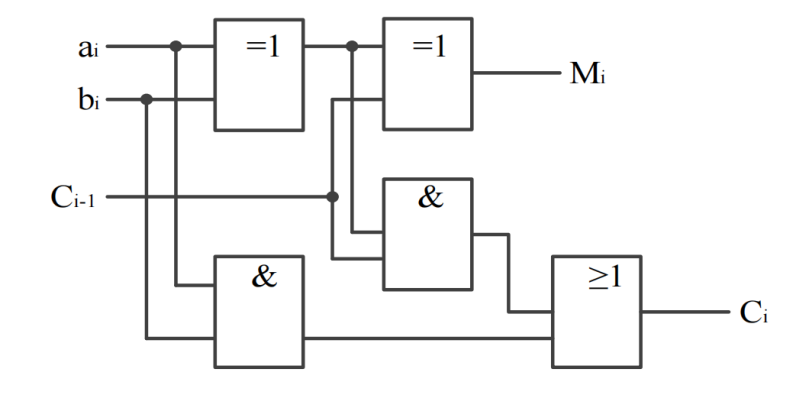


图2-3 全加器逻辑单元

可以看出进位 Ci来自两部分：第一部分是本位，可用表示，此部分和低位没有联系；第二部分与低位有关，其中可用表示。本位相加可能产生进位，本位的进位输出数值便是下一个全加器进位输入，进位这样一层层的传递会增加浮点加运算器的计算时间。所以在并行加法器中对于进位选择了并行进位链，并行进位链设计起来复杂，但是运行速度较快，以四位二进制数并行加法器为例，对于每一位的进位可表达为如式(2-6)所示。

(2-6)

将并行进位链表达式(2-6)转换成相应的逻辑图，其结构如图2-4所示。

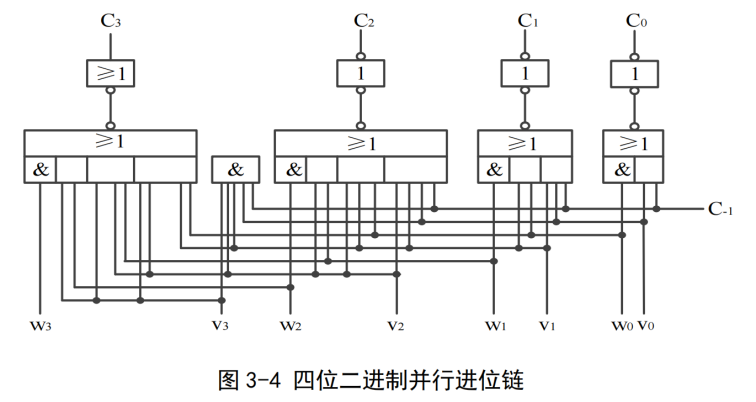


图2-4 四位二进制并行进位链

## 乘法

浮点乘法器的运算是比较消耗硬件资源的，尤其是浮点尾数的相乘，并且乘积结果可能是浮点数尾数宽度的两倍，本文对比一些乘法器的设计算法，如 Both 算法其乘运算是串行进行所以比较耗时，考虑到在浮点数运算器的运算速度，以及器件结构设计的难易程度，资源消耗等方面，所以本文设计的单双精度浮点数乘法运算器选择了阵列乘法器。

假设有两个规格化的浮点数、，， 为符号位，，为阶码，为偏移量，，为尾数其标准的表达式(2-7)所示。

(2-7)

两个浮点数相乘的运算结果的表达式如式(2-8)所示。

(2-8)

浮点数乘积运算结果的符号位，正数与正数相乘为正，负数与负数相乘得正，正数与负数相乘得负，其符号位求取逻辑符合异或运算。对于的乘运算结果并非规格化浮点数，假设乘的运算结果经过规格化处理后为，其可以表示为式(2-9)。

(2-9)

通过两个浮点数的乘积计算过程，可以总结出浮点数乘的运算可分为以下几步骤：

(1)分离浮点数：将浮点数的符号位，阶码位和尾数位进行分离，为后续的运算做准备；

(2)符号位异或：两符号进行异或运算，求出运算结果的符号位；

(3)阶码相加：计算出浮点数的实际指数值，添加双符号位，将指数值转换成补码，然后进行相加；

(4)浮点尾数相乘；

(5)规格化：运算结果可能需要右规1位；

(6)指数溢出判断：对相加的指数进行溢出判断，若发生溢出，则溢出标志位进行标志；

(7)尾数舍入：浮点数尾数乘积结果可能为尾数宽度的两倍，则必须对乘积结果进行舍入。

浮点数的乘积运算过程如图2-5所示。

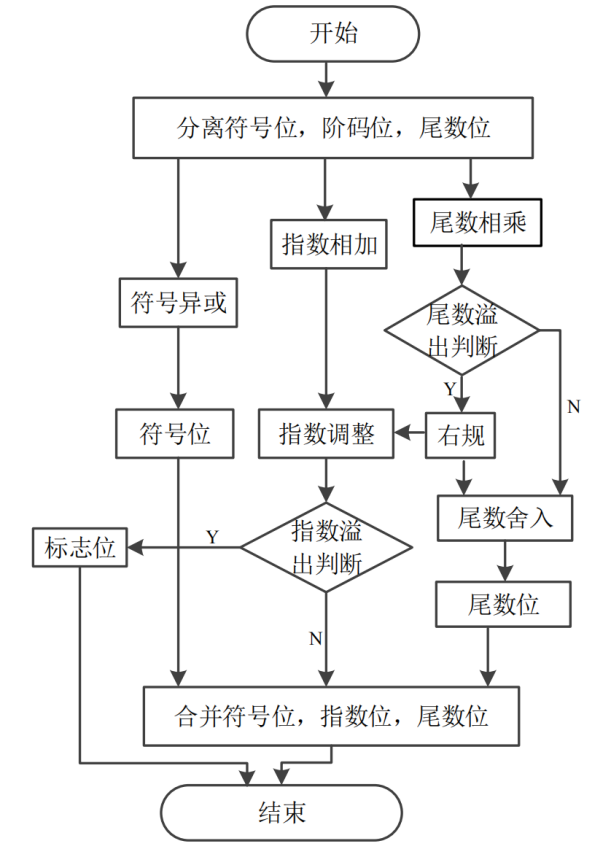


图2-5 浮点数乘法运算过程

假设两个四位二进制数与不含符号位且二者都为正，，，两二进制数手工相乘如图2-6所示。

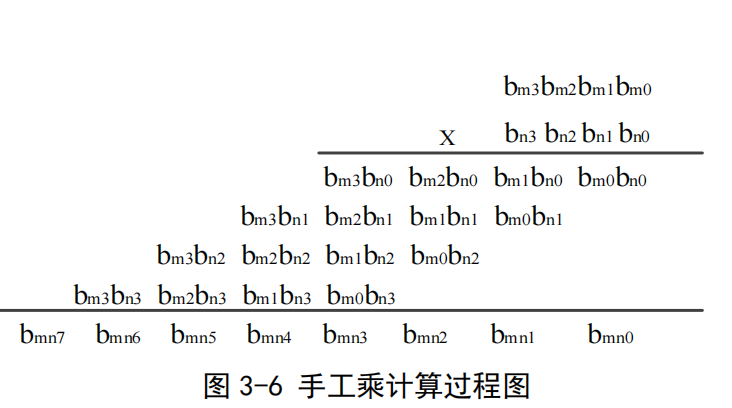


图2-6 手工乘法计算过程图

四位无符号二进制数和，假设为被乘数，为乘数，阵列乘法器的每一行将乘数数据从上往下由低位向高位一位一位的送入乘法器，且各行依次向左错开一位，则组成的每一斜列将被乘数数据从左往右由高往低一位一位的送入乘法器。其四位二进数和的绝对值阵列乘法器结构图如图2-7所示。

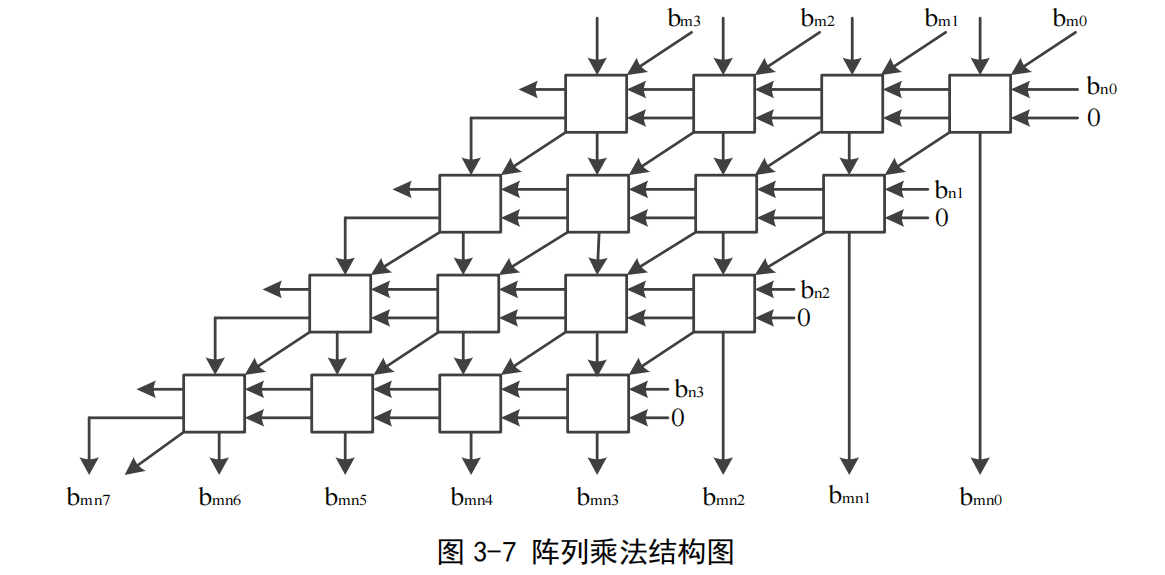


图2-7 阵列乘法结构图

图2-7就是绝对值乘的阵列乘法器的结构，对于阵列乘法器中的矩形方框以及方框周围的箭头将画图作介绍，如图2-8所示。

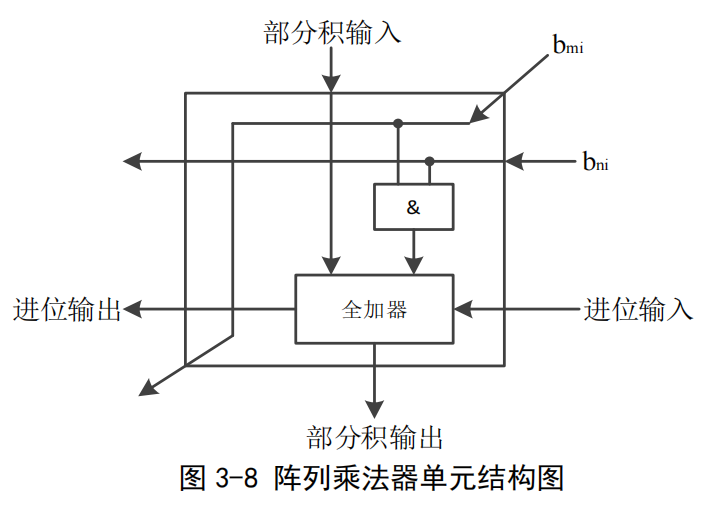


图2-8 阵列乘法器单元结构图

图2-7第一行的阵列乘法器同样采用图2-8的结构设计的方式输入，部分积输入的端没有数值所以 0 作为输入，同样在最右边的进位输入也是一样以 0 作为输入，其次阵列乘法器的其它部分的设计是以被乘数作为斜列输入端，以乘数为每行的起始输入端，以及乘积结果的输出端，另外还有计算过程中参与运算的进位数据输入输出端、以及部分积输入输出端，这些结构便构成了一个完整的阵列乘法器。浮点数乘法器选择阵列乘法器，加法器的数量有点多，不需要通过重复加与移位的运算步骤，因此乘法的速度得到了很大的提高。

## 除法

假设有两个规格化的浮点数、，， 为符号位，，为阶码，为偏移量，，为尾数其标准的表达式(2-10)所示。

(2-10)

若、分别为被除数和除数，两个浮点数相除的结果如式(2-11)所示

(2-11)

对于两浮点数和 X2 的除的运算结果并非规格化浮点数，需根据具体的数值做调整才可成为规格化浮点数，其次是浮点数除的运算结果的符号，和乘结果所得的符号原 则一样，正数与正数相除为正，负数与负数相除为正，正数与负数相除为负。假设规格化浮点数和 除的结果为，则的表达式如式(2-12)。

(2-12)

通过两浮点数除的计算过程，可以得出浮点数除的运算步骤可以分为以下几个步骤 ：

(1)检查除数：检查除数是否为零；

(2)分离浮点数：对浮点数的符号位，阶码位和尾数位进行分离，为后续的运算做准备；

(3)符号位异或：两符号进行异或运算，求出运算结果的符号位；

(4)阶码相减：计算出浮点数的实际指数值，添加双符号位，将指数值转换成补码，对指数相减同浮点数尾数相减一样处理，然后进行相加；

(5)浮点尾数相除；

(6)规格化：运算结果可能需要左规 1 位；

(7)指数溢出判断：对相减的指数进行溢出判断，若发生溢出，则溢出标志位进行标志；

(8)尾数舍入：对浮点数尾数除的结果进行舍入处理。

浮点数除的运算过程如图2-9所示。

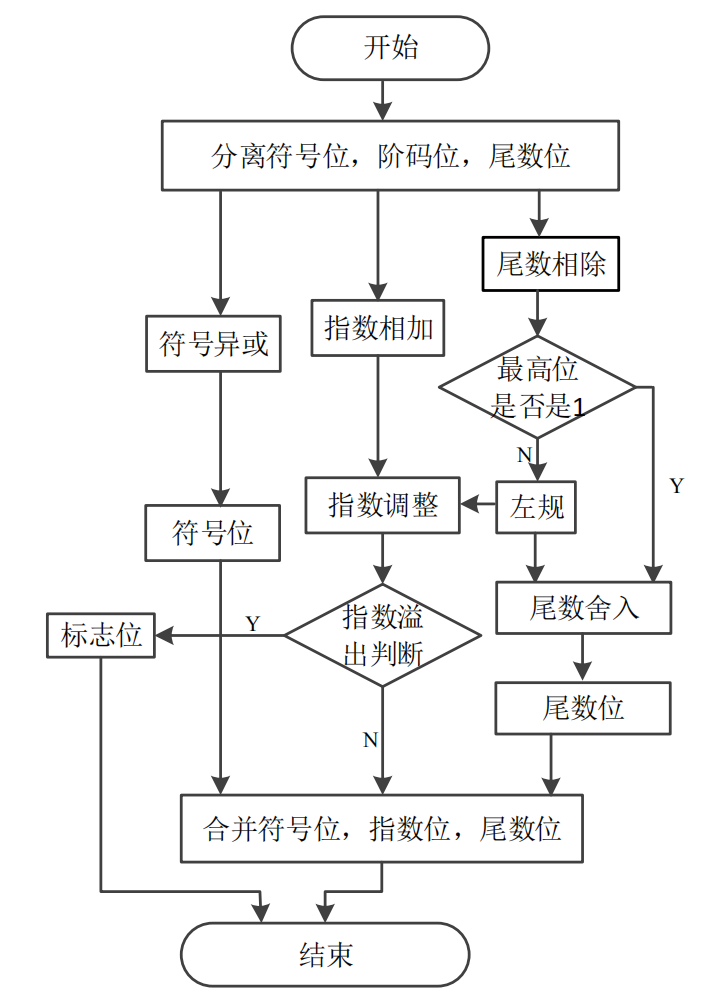


图2-9 浮点数除法运算过程

在浮点数尾数除法中，浮点数的符号位是单独进行处理，商值的符号是由两浮点数的符号通过异或运算得到的，对于浮点数的商值是由两浮点数的尾数的绝对值相除得到的。浮点数尾数的原码除法过程中对余数的处理方式有两种方法：第一种是恢复余数法；第二种是加减交替法。本文所设计的浮点数除法器选用的是加减交替算法。

阵列除法器的内部对数据的处理采用并行计算，相比较前期的串行除法器，阵列除法器与串行除法器各有优劣，阵列除法器在处理数据时运算速度是非常令人满意，但比较消耗硬件资源，其次阵列除法器的内部控制线路也比较少。阵列除法器还存在多种类型，比如加减交替法阵列除法器、补码阵列除法器等。本文所设计的浮点数除法器选用的是加减交替法阵列除法器。

加减交替法有一下两条运算规则：

(1)余数值的符号为正时，商值上1，余数整体向左移一位或者除数右移一位，然后减去除数得到下一个余数；

(2)余数值的符号为负时，商值上0，余数整体向左移一位或者除数右移一位，然后加上除数得到下一个余数。

加减交替法作除法举例：

，，求。

[x]补=0.10101001，[y]补=0.111，[-y]补=1.001。

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | 0.1 0 1 0 0 1 |  |  | 被除数x |
| +[-y]补 | **1.**0 0 1 |  |  | 第一步减去除数y |
|  | 1.1 1 0 0 0 1 | ＜0 |  | 余数为负，商0，下一步做加法  除数右移1位后加 |
| +[y]补 → | **0.0** 1 1 1 |  |  |
|  | 0.0 0 1 1 0 1 | ＞0 |  | 余数为正，商1，下一步做减法  除数右移2位后减 |
| +[-y]补 → | **1.1 1** 0 0 1 |  |  |
|  | 1.1 1 1 1 1 1 | ＜0 |  | 余数为负，商0，下一步做加法  除数右移3位后加 |
| +[y]补 → | **0.0 0 0** 1 1 1 |  |  |
|  | 0.0 0 0 1 1 0 | ＞0 |  | 余数为正，商1 |

故得商，余数。

在阵列除法器中，最基本的单元是可控加/减法(CAS)单元，其输入端和输出端分别各四个，存在一条输入控制端线，当为 0 时，CAS 执行加法运算；当为 1 时，CAS 执行减法运算，其次阵列除法器在运算时有些 CAS 还需处理符号位的运算。假设四位二进制浮点数，，要求，，是被除数，是除数，图 2-10 所示为可控加/减法(CAS)单元的逻辑结构图。

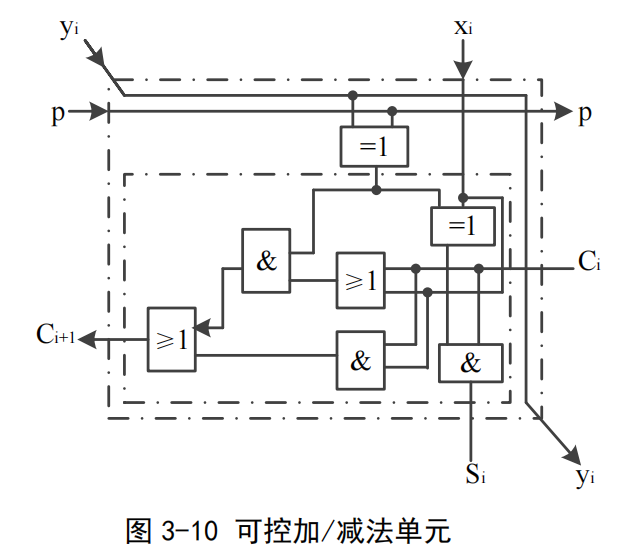


图2-10 可控加/减法单元

图2-10中为进位输入，为进位输出，为部分余数。对于绝对值相除的阵列除法器的整体构造，以四位二进数和为例，其中为被除数，为除数举例说明，被除数的数值一位一位的从左往右由高位到低位向阵列除法器第一行和最右边斜边上的各可控加/减法(CAS)单元竖直提供数据；除数也是一位一位的从左往右由高往低从第一行从对角线方向向阵列输入数据，这样余数不用移动只需右移除数，如同手工除法计算，商值的产生是由阵列最左边对角线上每一行可控加/减法(CAS)单元的输出进位 确定；余数是由阵列的最后一行产生，其次在阵列除法计算过程中，需要做 和的运算，对于减运算转换成补码[|X|]补+[-|Y|]补实现。

浮点数尾数绝对值阵列除法器，第一步是比较被除数和除数 的大小，通过运算然后检查两数的差值的符号位，即余数的符号位，以此来比较两数的大小，所以控制端值应为1，用减法完成[|X|]补+[-|Y|]补，并且通过两数相减的符号进位数值来确定商值，因此当商值为0时，表示被除数大于除数；当商值为1时，表示被除数小于除数。商值连接到下一行的控制端 P，有控制端决定本行做什么运算，即当控制端时，做加法运算；当控制端 时，做减法运算。如此反复下去，直至商值达到要求的精度为止。

对于绝对值除的阵列除法器如图 2-11 所示。

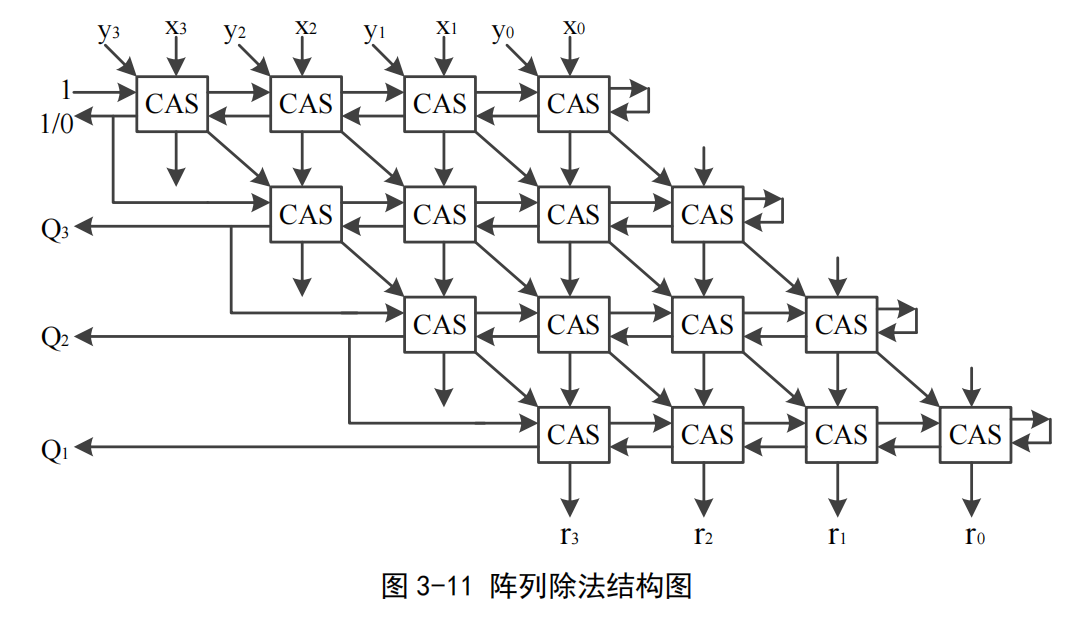


图2-11 阵列除法结构图

# 低复杂度矩阵求逆算法

对于阶数为M的方阵，若采用传统的矩阵求逆算法，通常需要耗费 的计算复杂度；对于大规模MIMO系统，这里的M值将会变得很大，所以精确矩阵求逆运算的复杂度将变得难以承受。因此，即便是相对来说已经是低复杂度的线性检测算法，如何降低其在大规模MIMO系统中的求逆运算的复杂度成为了亟待解决的核心问题。

本文调研了传统精确求逆算法以及几种矩阵求逆的低复杂度算法。

## QR分解法

QR分解求逆法是将待求逆的矩阵分为一个上三角矩阵和一个酉矩阵,该过程可以使用Gram-Schmidt正交化来实现。设待求逆的矩阵为：

 (3-1)

QR分解即将矩阵分解为:

 (3-2)

由于是酉矩阵，其共轭转置矩阵即其逆矩阵,所以只需对上三角矩阵求逆。基于Gram-Schmidt正交化的QR分解算法见表3-1，其中表示矩阵的第i行第j列元素,表示矩阵的第i列。输出矩阵即所求酉矩阵, 即所求上三角矩阵。

为求出的逆矩阵，还需要对上三角矩阵进行求逆操作。在此，表3-2给出下三角矩阵的求逆算法流程，并分析算法所需的操作数。由于是上三角矩阵，可以通过求取将矩阵化为下三角矩阵，再将求取转置来求出。

表3-1 基于Gram-Schmidt正交化的QR分解法

Table 3-1 QR decomposition algorithm based on Gram-Schmidt orthogonalization

|  |
| --- |
| **输入：** |
| for i = 1 : col    for j = i + 1 : col      end for |
| 输出 |

表 3-2 下三角矩阵求逆算法   
Table 3-2 Lower triangular matrix inversion algorithm

|  |
| --- |
| 输入: |
| for end for  end for |
| 输出 |

## LU分解法

分解是指将待求逆的矩阵分解为一个单位下三角矩阵和一个上三角矩阵,即:

(3-3)

由于是两个三角矩阵,在进行LU分解后还需要根据表3-2对矩阵与矩阵分别求取逆矩阵。

表3-3给出了分解的算法流程,其中与分别表示矩阵与矩阵的第行第列,表示矩阵的第行。

之后还需要利用表3-2的算法对两个三角矩阵求取逆矩阵后再相乘才可得到矩阵.

表 3-3 LU 分解的算法流程

Table 3-3 Algorithm flow of LU decomposition

|  |
| --- |
| 输入: |
| for col -1  for col  reci  reci  end for  end for |
| 输出 |

## Sherman-Morrison求逆法

Sherman-Morrison公式[30]是一个用于计算矩阵求逆的重要公式，它适用于特定情况下的矩阵。该公式的一般形式如下：

(3-4)

其中A是可逆方阵，x和y是列向量，并且不等于零。Sherman-Morrison公式允许通过已知矩阵A的逆和两个列向量x和y来计算矩阵 的逆，而无需直接计算 的逆。

## QR,LU和SM求逆算法对比

文章[34]对以上三种求逆算法的复杂度进行总结与对比。QR分解法、LU分解法的输入是Gram矩阵，而SM求逆法的输入是矩阵H，因此在计算QR分解法、LU分解法的复杂度时还要计算求取Gram矩阵的复杂度。使用三种算法分别处理一个8×8规模矩阵，矩阵求逆算法操作数如表3-4 所示。

表 3-4 LU 矩阵求逆算法等效操作数

Table 3-4 Equivalent computational operation number of matrix inversion algorithms

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 求逆算法 | 实数乘法 | 实数加法 | 实数求倒 | 实数开方 |
| QR分解法 | 5367 | 4928 | 16 | 8 |
| LU分解法 | 4866 | 4512 | 24 | - |
| SM求逆法 | 4040 | 3320 | 2 | - |

从表3-4中可以看出，SM求逆法具有最低的复杂度。

## Neumann级数展开

为了平衡计算复杂度和信号检测性能之间的关系,等在文献[26]中基于单载波频分复用(Single-Carrier Frequency-Division Multiple Access, SC-FDMA)的大规模MIMO系统中,提出了利用Neumann级数展开,将矩阵求逆作如下简化:

|  |  |
| --- | --- |
|  | (3-5) |

其中,当时,,级数收敛。通过分解正规化格拉姆矩阵,其中为矩阵的对角阵,为空心矩阵,将式(3-5)中的由取代可以表示为:

|  |  |
| --- | --- |
|  | (3-6) |

如果,则式(3-5)级数收敛。Neumann级数展开的核心思想在于可以展开式(3-6)的前项即可较为精确地近似表示。具体地,计算前项如下:

|  |  |
| --- | --- |
|  | (3-7) |

对较小的值,Neumann级数展开能以较小的复杂度近似计算。比如,当时,得到,相对于MMSE精确矩阵求逆时的计算复杂度,其计算复杂度为;当时,得到,其计算复杂度为。因此,能够以较低的计算复杂度获得一个二阶的Neumann级数近似;当时,得到。当展开级数时,Neumann级数展开的计算复杂度为,与MMSE精确矩阵求逆的计算复杂度相同,随着用户数呈立方增长。

为了简化Neumann级数展开算法在展开级数时的计算复杂度,文献[27]利用的主对角占优的特性,提出了一种改进后的Neumann级数展开算法。令,其中中的每个非对角元素的模都小于和中的非对角元素。换言之,当时,中的每个非对角元素的模逐渐接近于0。

基于上述的分析,当展开级数时,通过修改和简化后的Neumann级数展开可以表示为:

|  |  |
| --- | --- |
|  | (3-8) |
|  | (3-9) |

其中,表示矩阵的近似矩阵,表示的补偿矩阵。由于改进后的Neumann级数展开算法仅采用了信道矩阵的主对角和近对角元素来进行矩阵求逆,则矩阵求逆操作的计算复杂度减少为。

## Cholesky分解

Cholesky分解是一种将正定对称矩阵分解为下三角矩阵和其转置的方法。对于一个正定对称矩阵W，Cholesky分解可以表示为：

|  |  |
| --- | --- |
|  | (3-10) |

其中W是正定对称矩阵，L是一个具有正的对角线元素的下三角矩阵[28]，表示L的共轭转置。通过Cholesky分解，可以将原始矩阵W的求逆问题转化为下三角矩阵L的求逆问题，这可以更容易地处理。

为正定的赫米特矩阵,采用Cholesky分解矩阵求逆方法的实现过程为:

1)由Cholesky分解得到下三角矩阵及其共轭转置矩阵;

2)对下三角矩阵进行矩阵求逆计算,得到其逆矩阵;

3)由矩阵求得的逆矩阵。

其中L的求法如下：

|  |  |
| --- | --- |
|  | (3-11) |

由式(3-11)可得：

|  |  |
| --- | --- |
|  | (3-12) |

整理式(3-12)可得递推式：

|  |  |
| --- | --- |
|  | (3-13) |

## CSM算法

本文[31]针对大规模MIMO上行链路系统进行了研究，特别关注了Cholesky分解和Sherman-Morrison公式。在此基础上，提出了一种低复杂度的CSM检测算法，可以以 ()的复杂度实现MMSE检测算法的性能。此外，借助CSM检测算法解决复杂矩阵求逆问题的思想还可以应用于无线通信中的其他涉及矩阵求逆的信号处理问题。

### 具体算法

通过将Cholesky分解和Sherman-Morrison公式结合使用，可以迭代地求解大矩阵的逆，从而降低计算复杂度。

利用矩阵的正定对称性,采用Cholesky分解对矩阵进行分解,并利用Sherman-Morrison公式迭代求逆,避免直接求逆,从而降低了复杂度。

对矩阵进行Cholesky分解可得:

|  |  |
| --- | --- |
|  | (3-14) |

其中为下三角矩阵,对式(3-14)两边取逆,可得:

|  |  |
| --- | --- |
|  | (3-15) |

将的求逆运算转换为矩阵的求逆运算。而矩阵求逆的运算可由Sherman-Morrison公式迭代求得,迭代过程描述如下:

假设矩阵为可逆方阵,为列向量,有,同时可逆。则有Sherman-Morrison公式:

(3-16)

为了应用Sherman-Morrison公式降低计算的复杂度,需对下三角矩阵进一步分解:

(3-17)

其中,为由矩阵的对角元素组成的对角矩阵,为将矩阵的对角元素用0取代后的矩阵。

将式(3-17)展开如下:

(3-18)

其中,为单位矩阵的第行向量,为矩阵的第列向量,并令,可知是下三角矩阵,其行列式不为零,可得矩阵可逆,满足式(3-16)的应用条件。由式(3-16)可得的逆为:

(3-19)

其中,,再对进行求逆:

(3-20)

其中,。依次类推,直到计算得的逆为:

(3-21)

其中,。对角矩阵的逆求解简单，通过次的迭代运算求得。最后将求得的代入式(3-15),可求得。

### CSM与Neumann级数对比

本节分析算法计算复杂度中所需要的乘法运算。由式(3-14)将进行Cholesky分解,得到下三角矩阵。根据文献[32],Cholesky分解能够快速分解并通过FPGA硬件实现,所以忽略这部分的计算复杂度。式(3-19)和式(3-20)的分子上的,其列向量和行向量至少有一半以上的元素是0,根据稀疏矩阵和稀疏向量的特性,可以计算得到的计算复杂度为,然后经过次的迭代运算,得到的复杂度为。由此可得CSM检测算法的复杂度为。

诺依曼级数近似算法[33]和CSM检测算法的复杂度对比如表3-5所示。经典MMSE检测算法的复杂度是,从表3-5可以看出,当时,诺依曼级数近似算法能够把复杂度从降至。但在时,诺依曼级数近似算法复杂度都是,这就意味着它只能减少边缘的复杂度。而CSM检测算法能够把复杂度从降到,并且大规模系统的维数越大,降低复杂度的效果越佳。

表3-5 计算复杂度的对比

Table 3-5: Comparison of Computational Complexity

|  |  |  |
| --- | --- | --- |
| 的取值 | 诺依曼级数近似算法 | CSM 检测算法 |
| 2 |  |  |
| 3 |  |
| 4 |  |
| 5 |  |

# 低复杂度矩阵SVD算法

## Householder变换与QR分解计算SVD

一个矩阵(其中)的奇异值分解形式如下：

(4-1)

其中和两者相互正交且为酉矩阵，

|  |  |
| --- | --- |
|  | (4-2) |

其中表示奇异值，最后行只包含零项。

QR分解是将矩阵分解成一个标准的正交矩阵Q与一个上三角矩阵R，如下公式表示：

|  |  |
| --- | --- |
|  | (4-3) |

根据文献[40]，对QR分解中的R矩阵进行SVD，则有：

|  |  |
| --- | --- |
|  | (4-4) |

在A的奇异值分解中，(是正交矩阵的乘积)，以及，所以计算出矩阵R，则可以计算出矩阵A的奇异值与奇异向量，在此将使用Householder变换来对A进行分解。

Householder变换又称镜像变换，是一种正交变换，经过多次的Householder变换可以将矩阵变换为上三角形式，是一种常用的QR分解的方法。

基于Householder变换的QR分解步骤如下：

1. 将矩阵A按列分块，得到

|  |  |
| --- | --- |
|  | (4-5) |

取单位向量：

|  |  |
| --- | --- |
|  | (4-6) |

得到Householder变换矩阵

|  |  |
| --- | --- |
|  | (4-7) |

将左乘A，则有

|  |  |
| --- | --- |
|  | (4-8) |

1. 将第一步得到的

|  |  |
| --- | --- |
|  | (4-9) |

取

|  |  |
| --- | --- |
|  | (4-10) |

则

|  |  |
| --- | --- |
|  | (4-11) |

得到

|  |  |
| --- | --- |
|  | (4-12) |

1. 依此类推，进行迭代，在第n次迭代后如下：

|  |  |
| --- | --- |
|  | (4-13) |

此时矩阵成为矩阵，其中R为上三角矩阵，即所求 QR分解的R矩阵。

由于Householder矩阵的性质，设，对其求共轭有，其中Q包含了一个酉矩阵的前n列是标准正交的，这样就完成了QR分解。

## 随机奇异值分解

随机奇异值分解(Randomize Singular Value Decomposition，RSVD)是一种用于加速奇异值分解的近似算法，它通过对原始矩阵进行随机采样，然后再较小的采样矩阵上进行奇异值分解，从而近似地获得原始矩阵的奇异值和奇异向量。在大规模数据或高维矩阵的情况下能显著地减少计算和存储开销。

其原理可以如下表示：

对于给定的任意矩阵，其RSVD可以分为两步：

第一步：

1. 生成一个高斯随机的矩阵：

|  |  |
| --- | --- |
|  | (4-14) |

1. 构建矩阵Y为矩阵A与高斯矩阵的乘积：

|  |  |
| --- | --- |
|  | (4-15) |

1. 对Y进行QR分解，获得一个正交的矩阵与一个上三角矩阵:

|  |  |
| --- | --- |
|  | (4-16) |

第一步示意图可以如下表示：

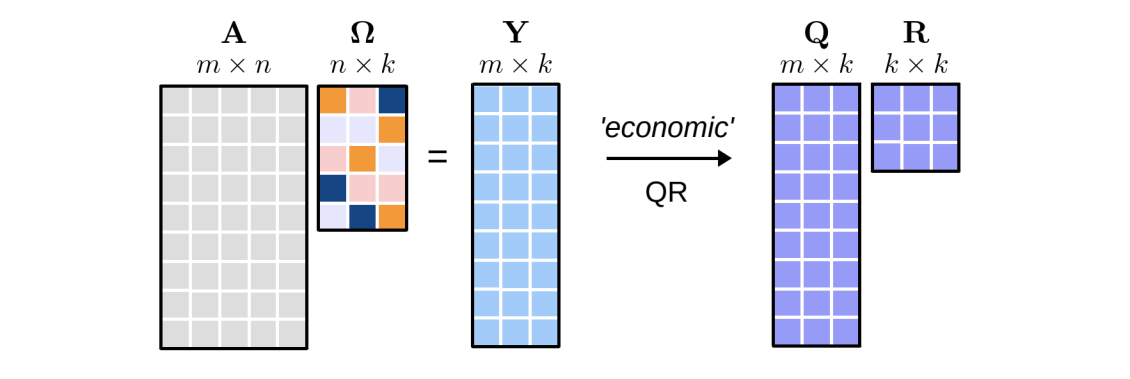


图 4-3 随机奇异值分解第一步示意图

第二步可以分为两小步，具体为：

1. 通过与进行矩阵相乘得到一个大小为的新矩阵：

|  |  |
| --- | --- |
|  | (4-17) |

1. 对矩阵进行奇异值分解，即

|  |  |
| --- | --- |
|  | (4-18) |

其中，矩阵的行数为，相比于对矩阵直接进行奇异值分解，这里对相对小一点的矩阵就显得更加节约计算。

需要注意的是，矩阵的奇异值和右奇异向量同时也是矩阵的奇异值和右奇异向量。

随机奇异值分解的第三步是计算矩阵的左奇异向量构成的矩阵，即：

|  |  |
| --- | --- |
|  | (4-19) |

其中，是由矩阵的左奇异向量构成的矩阵。

其第二步与第三步示意图可以如下表示：

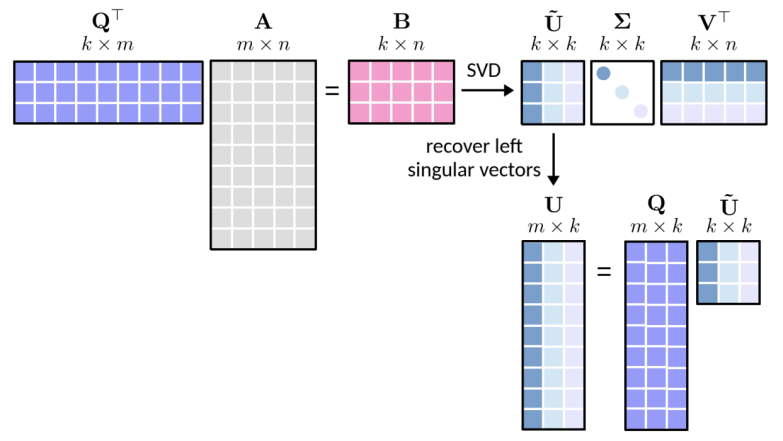


图 4-4 随机奇异值分解第二步与第三步示意图

其概括实现步骤如下：

(1) 先从原始矩阵随机选择一些列构成一个较小的采样矩阵；

(2) 再对采样矩阵进行QR分解，得到正交矩阵Q和上三角矩阵R；

(3) 然后将上三角矩阵R进行截断得到R’，此时可以理解为对原始矩阵的低秩近似。

(4) 再对R’进行SVD，得到近似的奇异值和奇异向量重构近似的矩阵表示。

随机化SVD是一种近似的SVD算法，其近似的程度取决于选择和截断的上三角矩阵的大小。由于其QR分解提供了原始矩阵的低秩近似，且对上三角矩阵进行了截断，有效地降低了运算的维度，提升计算效率。

## Lanczos迭代SVD

Lanczos迭代SVD通过迭代构造一个三对角矩阵，然后对这个三对角矩阵进行SVD分解，从而近似地获得原始矩阵奇异值和奇异向量的近似解，这种算法的一些工作最早可以追溯到Golub和Kahan的文章[39]，其中展示了如何通过Lanczos双对角化来计算奇异值。

一个矩阵()的SVD有如下形式：

|  |  |
| --- | --- |
|  | (4-20) |

其中和两者相互正交且为酉矩阵，，其中，为奇异值，和分别为矩阵A的左奇异值和右奇异值。

由式(4-20)中得，对于有：

|  |  |
| --- | --- |
|  | (4-21) |

其中，以及，当的时候认为是部分SVD(partial SVD)。

将其拆分，可看作：

|  |  |
| --- | --- |
|  | (4-22) |

对于一些起始向量(以及)，这个过程为Krylov子空间建立了标准正交基，

|  |  |
| --- | --- |
|  | (4-23)    (4-24) |

可以得到：

|  |  |
| --- | --- |
|  | (4-25)  (4-26) |

由可知与在Krylov子空间中列正交，残余向量满足，以及，是第m个适当尺寸的轴向量，以及

|  |  |
| --- | --- |
|  | (4-27) |

是一个上双对角矩阵，A的奇异值与奇异值相似，因此仅需对进行SVD即可得到与A相似的奇异值分解结果。

其概括过程如下：

(1)选择初始向量作为迭代的起点；

(2)通过迭代计算，构建一个双矩阵，每次迭代过程如下：(i)将当前向量正交化于前面的迭代向量集合上；(ii)将正交化的向量与原始矩阵相乘，得到一个新的向量；(iii)将新的向量正交化与前面的迭代向量集合上；(iv)将正交化后的向量与原始矩阵相乘得到新的向量；……(反复迭代直到满足迭代次数或其他终止准则)；

(3)对迭代过程生成的双对角矩阵进行SVD，得到近似的奇异值和奇异向量；

(4)利用近似的奇异值和奇异向量近似重构矩阵。

Lanczos迭代SVD的关键在于迭代构造三对角矩阵和截断操作能转化为较小规模的三对角矩阵SVD计算问题，从而提高计算效率和可扩展性。

## Jacobi求解SVD

1846年数学家Jacobi提出的经典Jacobi方法用于求解实对称矩阵的特征值。它的核心思想是采用一些列的Jacobi平面旋转矩阵将对称阵变为对角阵

 (4-28)

把左右的Jacobi旋转矩阵乘起来就是特征向量组成的特征矩阵， 为特征值组成的对角阵。Jacobi希望每通过一个Jacobi旋转矩阵都能消去矩阵中的非对角元素。

Jacobi旋转矩阵定义如下：

 (4-29)

一个Jacobi旋转矩阵，对角线上只有第i行i列，j行j列为c，其余为1，未标出的元素均为0。表示消去元素在矩阵中的位置，其中被称为旋转角，我们的目的就是找到一个合适的，使得非对角元上的两个元素变为0。由于仅影响i,j 行列的元素，故写为二阶主子式来表示旋转变换过程：

 (4-30)

其中

 (4-31)

解得:

 (4-32)

当一次变换结束后，和 同时为0，而矩阵的非对角元素的Frobenius norm的平方和将减少

 (4-33)

其中表示取A的对角线元素组成的对角矩阵。Frobenius norm的定义为：

 (4-34)

对于非对称矩阵，可以通过如下方法将其构造为对称矩阵：

 (4-35)

通过Jacobi旋转能将A矩阵获得其特征值，进一步能够求取其奇异值分解。有更进一步的算法，如单边Jacobi算法能加速其计算。

# 面向极低精度的可变精度计算深化设计

## 可变精度计算设计概述

复杂度制约着性能的进一步提升。过去的研究主要集中在降低复杂度的特定算法上。然而，还有一个很少被讨论的观点，那就是计算的精度。一般来说，更高的精度肯定会带来更好的性能，但性能会收敛。因此，当计算精度相对较高时，以性能换取复杂性是值得的。

浮点数是一种在计算机中存储数据的相互技术，最著名的标准是IEEE-754。由于存储容量和计算能力的限制，需要对精度较高的数字进行截断，以实现有限位存储。在此过程中引入了误差，称为舍入误差。了解误差的统计特征以及误差在计算过程中的传播方式非常重要。在[19]中，低精度存储引入的误差的概率分布函数(PDF)采用随机方法推导，在存储精度不太低的情况下易于实现。舍入误差的传播在[20]中进行了分析，但使用了高斯假设，这是一个没有证明的近似值。在[21]中使用随机模拟分析了不同算子的舍入误差，但不包括理论推导。在[22]中，以误差PDF的结论研究了定点乘法的舍入误差。

为了在精度和复杂性之间进行权衡，引入了不同精度的计算。文献[23]推导了矩阵逆的舍入误差统计模型。在[24]中，研究了精度可调的SVD。然而，所有这些方法都是基于算法的，没有考虑计算精度对复杂度和误差的影响。在通信系统中，精度和复杂性之间的权衡也是一个热门话题。在[25]中研究了计算复杂度和比奈奎斯特信号更快的性能之间的权衡。

## 下一阶段优化方向

### 指标设计优化

可变精度计算本质上是一个计算精度与计算复杂度之间的优化问题，因此计算精度指标的设计至关重要。从定性的角度看，计算精度与计算误差之间显然存在负相关关系。但是定量地刻画两者之间的关系则相对困难。

最为传统和直接的指标是相对误差指标，其计算简单开销小，但是在遇到计算结果为0的情况时会出现“除0”问题导致指标变成无穷大。又因为可变精度计算使用随机数学分析计算过程，因此所有的指标设计都是考虑计算误差的统计特征。若考虑使用矩作为统计特征，则会因为“除0”问题导致各阶矩都变成无穷大；若考虑使用置信区间作为统计特征则可以有效避免这个问题。但是相比于矩，置信区间不仅难以获得闭式，同时需要知晓误差PDF，因此在已有的成果中使用了矩作为统计特征，同时配合一些鲁棒性设计规避“除0”问题。下面是对于不同统计特征的总结。

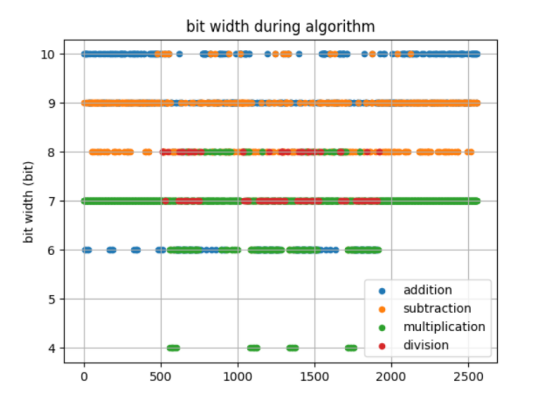
表5-4 不同统计特征的总结

|  |  |  |  |
| --- | --- | --- | --- |
|  | 数学表达 | 优点 | 缺点 |
| 矩  (期望、方差) |  | 有闭式、无需PDF | 鲁棒性差(“除0”) |
| 置信区间 |  | 鲁棒性强 | 闭式难以获取，需要PDF |

### 尺度设计优化

在已有成果中，可变精度计算做到了算子级，这是最小的优化尺度，理论上具有最优的结论。但是考虑到实际工程需要，算子级可变精度设计的额外开销相对较大，同时性能增益相比尺度更大的设计有限。

在工程应用中，可变精度的优化尺度从小到大可以分为算子级、复数级、矩阵级、算法级。对于不同场景可以采用不同优化尺度，平衡可变精度收益与开销。



(a)算子级

文本

描述已自动生成

(b)矩阵级

表格

描述已自动生成

(c)算法级

图5-5 可变精度优化尺度示例

### 效用函数设计优化

在可变精度计算的优化问题中，需要建立计算性能与计算精度之间的量化关系。从定性的角度看，计算精度与计算性能之间正相关，但是两者的定量关系受具体的指标、计算方式等多因素影响，获取较为困难。已有成果中对该问题仅考虑了工程化实现，即根据计算性能与计算精度之间一一对应的关系，通过遍历的方式获得两者之间的定量关系。在本期项目中将考虑针对一些通信算法中的常见指标(和速率、BER等)推导计算性能与计算精度之间的定量关系，完善可变精度计算的理论框架同时，可以了解预编码矩阵不同元素对于最终通信性能指标的贡献。

### 超低精度(6-8bit)可变精度计算设计

对于绝大多数算法，直接采用超低精度计算必然无法满足性能需求。因此，考虑采用低精度迭代高精度思路，在计算和存储两个维度实现超低精度计算。

在已有成果中实现了可变精度的存储设计(eBFP)，其实现了浮点数浮动位的可变精度。但是eBFP设计的背景是考虑存内计算架构，因此将每一个块长度固定为8bit。在本期项目中，考虑进一步开发存储方式的灵活度，一方面优化每一个存储块的存储长度，另一方面考虑在极低位宽场景下动态分配指数位和浮动位位宽，实现有限位宽的最高效利用。

另一方面，一部分对精度要求较高的算法无法直接应用低精度计算，因此考虑设计迭代型算法设计，通过权衡迭代次数与计算精度优化算法性能。具体实现方式包括类似泰特展开的线性逼近以及尾数收敛等。

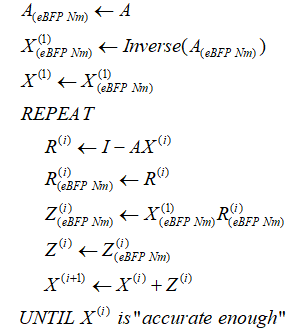


图5-6 迭代型算法设计示例

# 低精度预编码

在MIMO系统中，如果发射端能够获知信道的某些信息，就可以利用该信息对发射信号进行预处理以提高系统的传输速率和链路可靠性，这种利用发送端信道状态信息对发送信号进行与处理的技术称为预编码技术。

预编码技术可以按照编码设计方案分为线性预编码和非线性预编码两类，在多用户MIMO系统中，在基站发送端进行线性处理的发送信号方法称为线性预编码，主要的线性预编码方案有迫零(Zero-Forcing, ZF)预编码、最小均方误差(Minimum Mean Square Error, MMSE)预编码以及块对角化(Block Diagonalization, BD)预编码。非线性预编码是指在发送信号上添加设计好的预编码矢量来达到抑制用户间干扰的预编码方法，非线性处理方法主要是基于“脏纸编码(Dirty Paper Coding, DPC)”思想的各种预处理方案，尽管非线性预编码技术性能足够优秀，但是其实现条件较高，并且有着极高的计算复杂度，对于硬件实施和运维管理是不切实际的。随着硬件发展的放缓，为了提高运算的能力，在低精度的条件下进行预编码并减少误差变得十分必要。

## 低精度预编码

在多种线性预编码技术当中都需要进行矩阵的求逆运算或对矩阵进行分解，如ZF预编码矩阵及MMSE预编码矩阵，以及SVD预编码，这些矩阵运算在低精度环境下会产生误差，但现在少有直接对预编码技术在低精度下的探究工作，更多的是对矩阵运算(如求逆、QR分解等)的低精度探索及误差分析。

在1961年的一篇经典论文[44]中，Wilkinson对LU分解、Givens QR分解和Householder QR分解进行了舍入误差分析。他指出:“我们得到的边界在所有情况下都是严格的上界。一般来说，舍入误差的统计分布会大大减少相对误差中出现的关于n的函数。我们可以期望在每种情况下，这个函数都应该被一个不大于它的平方根的东西所取代，而且通常要小得多。”最近的概率舍入误差分析为Wilkinson的LU分解和其他基于内积的计算提供了严格的基础。

Yang, Fox和Sanders[45]考虑了一种混合精度的Householder QR分解算法，这种方法工作精度为以及内积精度为，他们得到的一个正态后向顺序误差界为，其中m和n分别为矩阵的行数和列数。Connolly[46]提出的随机误差方法更适合Householder QR分解算法，其算法得到的误差界能低至

在[47]中，给出了一种计算QR分解的随机化过程，该方法采用随机Gram-Schmidt过程，采用随机扫描降维技术，在舍入误差为均值为零的独立随机变量的假设下对其进行了分析，他们利用两种精度()的混合精度算法，其中用来计算投影，用来计算其他内容，其结果为，其中和为在该精度下计算得到的A的QR分解矩阵。

这些工作都聚焦在对矩阵QR分解的工作上，对于进一步在预编码技术上的应用还需进一步研究。

## 低复杂度预编码

许多算法通过提高预编码的算法复杂度来获得更高的计算准确率，但是在低精度的计算环境下，预编码的复杂度越高，计算机进行存储与计算的次数也随着增加，每一次数据存储和计算的损失不断累积，最终造成破坏性的误差。因此，高准确率但复杂的算法在低精度的计算环境下不一定能成为最优选择，需要探索较高准确率但计算复杂度低的算法进行预编码，这样能有效减少数据计算和存储过程中产生的损失。

文献[48]提出了一种基于Gram-Schmidt共轭方向迭代法的新型低复杂度近最优预编码器，将复杂度从降到了。其思路如下，对于一个迫零预编码器而言，其预编码矩阵如下表示：

(6-1)

其中为功率因子，为信道矩阵，对于要传输的信号与实际符号之间有如下关系：

(6-2)

其中

(6-3)

文中为了避免对矩阵求逆，故上式等价为线性方程的解：

(6-4)

由于大规模MIMO中信道矩阵行向量之间接近正交，那么是Hermitian矩阵以及正定矩阵且对角占优。基于这些特征，文章提出了基于Gram-Schmidt共轭方向迭代的方法，这是个有效地迭代过程，它最小化了二次函数：

(6-5)

可以得到最小化式(6-5)与求解式(6-4)是等价问题。这种提出的算法每次迭代具有较低的复杂度，并且能在计算K次后收敛，并且可以提前结束迭代获得接近精确的解，因此能作为替代计算的方法。

其具体算法如下：

|  |
| --- |
| 算法1：基于GSCD的预编码 |
| 1：输入：  2： 和  3：初始化：  4： 和  5：  6： 和  7：  8：当时执行循环  9：  10：  11：  12：  13：  14：结束循环  15：输出  16： |

其中为原始信号，为平坦瑞利衰落信道矩阵，这个算法避免了计算矩阵的逆，这种算法在大规模MIMO仿真中验证了其有效性，其展示了在不同迭代数量下，与基于NS的预编码的复杂度对比，如下表所示

表6-1 在不同迭代数量下与基于NS的预编码的复杂度对比

|  |  |  |
| --- | --- | --- |
| 迭代次数 | 基于NS的预编码 | 基于GCSD的预编码 |
| k=1 |  |  |
| k=2 |  |  |
| k=3 |  |  |
| k=4 |  |  |

文献[49]中为了避免信道矩阵SVD的高复杂度提出了一种基于主成分分析技术的秩-1近似分解算法。

对于大规模MIMO信道中矩阵SVD的分解，其有着高复杂度，这种秩-1近似分解算法将复杂度降至，具体算法如下：

|  |
| --- |
| 算法2：的低复杂度秩-1近似分解 |
| 1：输入：  2： 第k个用户信道矩阵  3：如果  4：  5：选择合适的整数和使得和  6：生成一个独立同服从标准正态分布的矩阵  7：  8： for 执行  9：  10：end  11：构建矩阵  12：对进行QR分解，  13：  14：对T进行SVD分解  15：  16：令，,  17：输出：最大奇异值和相应的左右奇异向量和。 |

在算法2中，为了计算矩阵A需要flops，在步骤12中QR分解和步骤14中的SVD都需要flops，构建矩阵和分别需要和flops，此外，对于整数和的选择，它们越大计算复杂度越高，近似精度也越高。在文中说明对于和对于多数应用来说是足够的，综上所述，可以得到算法2的复杂度为：

(6-6)

这种秩-1的近似算法进一步地降低了信道矩阵SVD的复杂度，其复杂度评价也比一些最先进的算法具备更低的复杂度，其仿真结果验证了其有效性。

## 非完美CSI建模

在发射端采用信道状态信息(CSI)的传输预编码可以进一步提高多输入多输出(MIMO)衰落信道的容量，但性能的提高很大程度上取决于对CSI的了解程度。接收端信道估计误差、CSI量化噪声和反馈延迟引起会导致非完全CSI。利用矩阵扰动理论可以分析非完全CSI条件下收发端阵列增益损失、子信道间干扰、信噪比(SINR)和CSI误差对线性预编码系统性能的影响。

### 非完美CSI建模过程[50]

假设是维度为的MIMO通道矩阵，并且对于任意的，都是独立同分布的，每一项都符合每个维度方差为0.5的循环对称复高斯分布(ZMCSCG)，即。如果和分别为接收端和发射端估计的信道，则可以将发射端和接收端的实际MIMO信道分别建模为

(6-7)

其中和分别为接收端和发射端信道估计的误差，它们的每一项，分别满足方差为和的循环对称复高斯分布。是噪声的方差，一般来说，。由于，可以得到，。

在接收端，通过矢量量化技术进行量化，然后再反馈到发射端。假设为量化误差的方差，则如果没有反馈延迟时间，则接收端得到的信道状态信息(CSIT)误差的方差为。当CSI反馈的延迟时间为，发射机具有与过时的信道信息。因此，给定，我们可以描述当前发射端的CSI如下：

(6-8)

其中为独立同分布的时变MIMO信道的时间相关系数。

发射端和接收端都做了信道估计，得到和，现在分别对和做SVD分解得到

(6-9)

于是发射端的预编码矩阵和接收端的滤波矩阵可以分别表示为

(6-10)

其中，是矩阵的秩，是满足的条件下分配给第个子信道的功率，是一个常量对角矩阵。和分别为和的前个列向量组成的矩阵。于是经过整个通信过程后接收端接收到的信号为

(6-11)

其中**，**是奇异值按递减方式排列的对角矩阵。由于整个通信过程可以被充分地对角化，由第个子信道传输的信号到达接收端后可以表示为

(6-12)

是第个子信道接收到的噪声。其中是矩阵的第列，是矩阵的第列。令，，，，，。因此得到

(6-13)

式中第一项是期望收到的信号，第二项是收到的噪声和干扰。再令，，，当时，表明发射端和接收端都知道准确的CSI。因此第个子信道接收到的信号功率和噪声与干扰功率之和的比为

(6-14)

假设和分别是第个子信道中发射端和接收端的波束赋形矢量，和分别是真实的信道矩阵的第个左奇异值向量和第个右奇异值向量。和的相关系数以及和的相关系数可以表示为

， (6-15)

对于给定的和，条件相关矩阵和可以写为

(6-16)

(6-17)

因此，接收信号和发射信号的功率系数可以用互相关系数写为

(6-18)

同理

(6-19)

于是就可以计算(6-14)式中的。

### 非完美CSI情况下的资源分配(数据位数和功率的分配)[51]

考虑一个具有发送时长限制的传输系统和一个在时隙内传输的位数据包。假设不同插槽的衰落信道状态是独立且同分布的。为了便于描述，假设时隙按降序排列，是初始时隙，是第二个时隙，……，是最后一个时隙，即。此外，在每个时隙中只能传输一个数据符号。数据传输采用根据使用的调制方案不同的速率。在不丧失一般性的情况下，我们假设有种调制阶数，每个符号的位数可以定义为，且。因此，在每个时隙上的可用传输比特数集被表示为。然后，将要服务的比特数和时隙处队列中的剩余比特数可以分别记为和。显然，属于集合。

假设是接收机接收到的噪声能量。并且和分别为在第个时隙中估计到的信道增益和真实的信道增益。接收信号的质量可以作为接收的信噪比(SNR)值来评估

(6-20)

其中第个时隙中的发射信号功率，由发射端发射信号的信噪比(txSNR)。

为了保证接收信号的准确度，实际信噪比必须不低于目标信噪比，即

(6-21)

其中是需要达到的误码率。

在不知道信道估计误差分布信息的情况下，利用椭球面近似提出了一个鲁棒调度问题；并且在知道信道估计误差分布信息的情况下，利用传输可靠性约束，提出了鲁棒调度问题。鲁棒性是指在不完全CSI下成功传输包的能力。

无法得到信道估计误差分布信息时，CSI的估计误差可以确定地建模为不确定集如下

(6-22)

是信道估计误差，是的上界。

构造该问题的目的是确定在每个时隙期间要传输的比特数，从而使预期的总传输功率最小化。利用鲁棒优化理论，鲁棒调度问题可以表述为一个随机规划问题如下

(6-23)

满足式(6-20)和(6-21)，并且还满足

(6-24)

(6-25)

式(6-25)表明在接下来的个时隙内必须传输不少于个比特。

由于被限定在范围内，故，即。上述优化问题可以重写为

(6-26)

满足式(6-24)，式(6-25)，并满足。。

当已知信道估计误差分布时，真实的信道增益依然可以建模为。用dB表示信道估计误差为

(6-27)

其中。服从均值为0，方差为的高斯分布。此外，和相互独立。

由于CSI的不确定性，存在一个时隙传输失败的概率。给定估计的信道增益，可以得到一个时隙的成功传输概率如下

(6-28)

其中为误差函数。

为了保证一个时隙的可靠传输，应满足以下传输的可靠性约束

(6-29)

为传输可靠性。由式(6-28)和式(6-29)可知，在某一个时隙中发射端要求的最小信噪比(txSNR)为

(6-30)

其中，是误差函数的反函数。

因此这个具有鲁棒性的调度问题可以建模为

(6-31)

满足式(6-24)、式(6-25)和式(6-29)。

在实际传输系统中，可以通过信道反馈获得当前CSI，但未来的信道状态是未知的。因此，经典的组合优化方法不适用于求解鲁棒优化问题(6-26)和(6-31)。从实现的角度来看，我们尝试使用动态规划(DP)来寻找优化问题的调度解决方案。

为了保证每一个时隙的数据都保证能可靠传输，当某个时隙发射端发送信号的信噪比低于信号可靠传输所需的目标信噪比时，该时隙发送的比特位需要被重新发送。

在式(6-23)中，在第个时隙中，发射端发射信号的信噪比为，利用DP原理，通过求解以下优化问题，可以得到优化问题(6-23)的解

(6-32)

满足以及

(6-33)

(6-34)

优化函数(6-32)中的第一项表示时隙处的txSNR，第二项表示未来时隙中的预期总txSNR。需要注意的是，由于整数位加载的限制，在剩余时隙中可以传输的最大比特数为。因此，如果是在当前时隙中传输的比特数，则应满足，以保证所有剩余的比特都可以在规定的时间内传输。为了便于描述，可以将传输容量和位加载的两个约束条件重写为(6-27)中定义的新集合。受不完全CSI的影响，传输的数据位可能无法被正确接收。因此，在时隙t处的传输失败的情况下，时隙处的剩余位为。另一方面，当位在时隙t处成功传输时，时隙处的剩余位为。数学上，时隙的剩余位为

(6-35)

此外，(6-34)表示在时隙的队列中剩余的位应该在未来的时隙内提供服务。因此，如果，或，则将发生中断，即含有个比特位的数据包不能在个时隙内传输。

同样地，对于优化问题(6-31)，在第个时隙中，发射端发射信号的信噪比为，那么优化问题(6-31)的解可以通过解如下优化问题得到。

(6-36)

满足式(6-29)、式(6-33)和式(6-34)。

总的来说，优化问题(6-32)和(6-36)可以分别用于描述没有信道估计误差信息和已知信道估计误差信息情况下的鲁棒分组调度问题。

# 总结与研究思路

面对日益增长的通信规模以及逐渐严苛的端到端时延要求，物理层波形设计所产生的计算实验已逐渐成为主要成分。在传统通信算法优化已经较为成熟的当下，优化计算为进一步降低计算时延提供了新的途径。传统的计算优化包括使用更低的精度进行计算，以获得在计算复杂度与计算性能之间的权衡。然而物理层波形设计对于精度及其敏感，使用传统方法暴力降低计算精度将对严重降低通信性能。

可变精度计算设计是一种根据计算重要性灵活调度计算资源的计算设计，是优化计算中的全新设计。而算子级可变精度计算则可以在计算复杂度与通信性能之间取得更优的权衡。为了进一步优化可变精度计算设计，以支持更低精度的计算，本项目的研究思路如下：

1. 探索全新的低精度存储设计。

IEEE754中最低的存储精度为16bit，为了支撑更低精度的计算，需要设计一种全新的低精度存储。根据传统的浮点数设计方法，在极低位宽下，存储范围与存储精度都将骤降，严重影响计算性能。因此本项目考虑结合定点数存储设计经验，设计可变存储范围的浮点数存储设计，释放表示存储范围所用的存储空间，将有限的空间用于更高精度的存储。为了支持该设计，需要设计一种可以在算法中级联的存储范围定义。

2. 设计低复杂度矩阵计算算子。

计算误差会随着计算次数的增加而扩大，因此对于低精度计算，控制总的计算次数显得尤为重要。对于高精度的复杂矩阵计算算子（如求逆、SVD），其结果精度在低精度下反而会被冗长的计算步骤而拖累，甚至得到比步骤少的低精度算子更差的结果。因此本项目考虑设计一种可以灵活权衡计算复杂度（计算次数）与计算性能的矩阵算子，以灵活匹配各种计算精度的需求。

3. 设计面向低精度计算的预编码算法。

除了需要优化矩阵算子以减少总计算次数，同样可以结合预编码算法设计优化，以减少整个算法的总计算次数，达到减小计算误差的目的。同时，本项目考虑研究预编码矩阵中不同位置元素对最终通信指标（如BLER）的贡献，以实现更加灵活的计算资源分配，设计对于误差不敏感的通信预编码算法。

# 参考文献

1. M. Hashemi, Khubaib, E. Ebrahimi, O. Mutlu, and Y. N. Patt, “Accelerating dependent cache misses with an enhanced memory controller,” in *Proc. ACM/IEEE 43rd Ann. Int. Symp. Comput. Architect. (ISCA)*, Seoul, Korea (South), Aug. 2016, pp. 444–455.
2. J. S. Kim et al., “Revisiting row hammer: An experimental analysis of modern devices and mitigation techniques”, in *Proc. ACM/IEEE 47th Ann. Int. Symp. Comput. Architect. (ISCA)*, Valencia, Spain, Jul. 2020, pp. 638-651.
3. A. Boroumand et al., “LazyPIM: An efficient cache coherence mechanism for processing-in-memory,” *IEEE Comput. Architect. Lett*., vol. 16, no. 1, pp. 46-50, Jun. 2017.
4. M. Kang, M. S. Keel, N. R. Shanbhag, S. Eilert, and K. Curewitz, “An energy-efficient VLSI architecture for pattern recognition via deep embedding of computation in SRAM,” in *Proc. IEEE Int. Conf. Acoust., Speech Signal Process. (ICASSP)*, Florence, Italy, May. 2014, pp. 8326–8330.
5. A. Shafiee et al., “ISAAC: A convolutional neural network accelerator with in-situ analog arithmetic in crossbars,” in *Proc. ACM/IEEE 43rd Ann. Int. Symp. Comput. Architect. (ISCA)*, Seoul, Korea (South), Aug. 2016. pp. 14-26.
6. V. Seshadri et al., “Ambit: In-memory accelerator for bulk bitwise operations using commodity DRAM technology,” in *Proc. 50th Ann. IEEE/ACM Int. Symp. Micro archit. (MICRO)*, Boston, MA, USA, Oct. 2017, pp. 273-287.
7. S. Angizi, Z. He, and D. Fan, “PIMA-logic: A novel processing-in-memory architecture for highly flexible and energy-efficient logic computation,” in *Proc. 55th ACM/ESDA/IEEE Des. Automat. Conf. (DAC)*, San Francisco, CA, USA, Jun. 2018, pp. 1-6.
8. S. Angizi, Z. He, A. S. Rakin, and D. Fan, “CMP-PIM: An energy-efficient comparator-based processing-in-memory neural network accelerator,” in *Proc. 55th ACM/ESDA/IEEE Des. Automat. Conf. (DAC)*, San Francisco, CA, USA, Jun. 2018, pp. 1-6.
9. J. Ahn, S. Yoo, O. Mutlu, and K. Choi, “PIM-enabled instructions: A low-overhead, locality-aware processing-in-memory architecture,” in *Proc. ACM/IEEE 42nd Ann. Inter. Symp. Comput. Architect. (ISCA)*, Portland, OR, USA, Jun. 2015, pp. 336-348.
10. J. Ahn, S. Hong, S. Yoo, O. Mutlu, and K. Choi, “A scalable processing-in-memory accelerator for parallel graph processing,” in *Proc. ACM/IEEE 42nd Ann. Int. Symp. Comput. Architect. (ISCA)*, Portland, OR, USA, Jun. 2015, pp. 105-117.
11. I. Pletea, Z. Wurman, Z. Or-Bach, and V. Sontea, “Monolithic 3D layout using 2D EDA for embedded memory-rich designs,” in *Proc. IEEE SOI-3D-Subthreshold Microelectron. Technol. Unified Conf. (S3S)*, Rohnert Park, CA, USA, Oct. 2015, pp. 1-2.
12. 薄列峰, 顾松庠, 陈彦卿等. 联邦学习：算法详解与系统实现, 北京：机械工业出版社, 2022.
13. M. Barthel, J. Rust and S. Paul, “Hardware implementation of basic arithmetics and elementary functions for unum computing,” in *Proc. 52nd Asilomar Conf. Signals, Syst., Comput., (ACSSC)*, Pacific Grove, CA, USA, Oct. 2018, pp. 125-129.
14. M. Bärthel, J. Rust and S. Paul, “Application-specific analysis of different SORN datatypes for unum type-2-based arithmetic,” in *Proc. IEEE Int. Symp. Circuits Syst., (ISCAS)*, Seville, Spain, Oct. 2020, pp. 1-5.
15. S. Knobbe, M. Bärthel, S. Paul and J. Rust, “Complexity reduction for sphere decoding using unum-type-II-based SORN-arithmetic,” in *Proc. 9th Int. Conf. Modern Circuits Syst. Technol.,* *(MOCAST)*, Bremen, Germany, Sep. 2020, pp. 1-4.
16. Gustafson, John, The End of Error: Unum Computing. BOCA RATON, Fla: Chapman and Hall/CRC, 2015.
17. Andrea Bocco, Yves Durand, and Florent De Dinechin, “SMURF: Scalar multiple-precision unum risc-v floating-point accelerator for scientific computing,” in *Proc. Conf. Next Gener. Arithmetic (CoNGA)*. New York, NY, USA, Mar. 2019, pp. 1-8.
18. J. A. Hittinger, et al, “Variable Precision Computing,” *Lawrence Livermore National Lab (LLNL)*, Livermore, CA, US, 2019.
19. IEEE Standard for Floating-Point Arithmetic, IEEE Std 754-2019, Jul. 2019.
20. F. Dahlqvist, R. Salvia, and G. A. Constantinides, “A probabilistic approach to floating-point arithmetic,” in *Proc. 53rd Asilomar Conf. Signals, Syst., Comput. (ACSSC)*, Pacific Grove, CA, USA, Nov. 2019, pp. 596–602.
21. F. Chatelin and M. C. Brunet, “A probabilistic round off error propagation model. application to the eigenvalue problem,” in *Reliab. Numer. Comput.*, M G Cox, and S Hammarlin, Eds. Oxford: Clarendon Press, Sep. 1990, pp. 139-160.
22. M. Fasi and M. Mikaitis, “Algorithms for stochastically rounded elementary arithmetic operations in IEEE 754 floating-point arithmetic,” *IEEE Trans. Emerg. Topics Comput.*, vol. 9, no. 3, pp.1451–1466, Mar. 2021.
23. M. Urabe, “Roundoff error distribution in fixed-point multiplication and a remark about the rounding rule,” *SIAM J. Numer. Anal.*, vol. 5, no. 2, pp. 202–210, Nov. 1968.
24. M. Tienari, “A statistical model of roundoff error for varying length floating-point arithmetic,” *BIT Numer. Math.*, vol. 10, pp. 355–365, Sep. 1970.
25. C. H. Wu and P. Y. Tsai, “An svd processor based on golub–reinsch algorithm for mimo precoding with adjustable precision,” *IEEE Trans. Circuits Syst. I: Reg. Papers*, vol. 66, no. 7, pp.2572–2583, Mar. 2019.
26. Wu Michael, Yin Bei, Wang Guohui, et al, “Large-Scale MIMO detection for 3GPP LTE: Algorithms and FPGA implementations,” *IEEE J. Sel. Top. Signal. Process.*, vol. 8, no. 5, pp. 916-929, Oct. 2014.
27. B. Kang, J. H. Yoon, J. Park, “Low complexity massive MIMO detection architecture based on Neumann method,” in *Proc. Int. Soc Des. Conf.* *(ISOCC)*, Gyeongju, Korea (South), Nov. 2015, pp. 293-294.
28. 陈晓东, 李世平, 何国强. 基于FPGA的Cholesky分解矩阵求逆. 现代雷达, 2019, 41(10): 58-61.
29. Ries F，De Marco T，Guerrieri R. “Triangular matrix inversion on heterogeneous multicore systems,” *IEEE Trans. Parallel. Distrib. Syst.*, vol. 23, no. 1, pp. 177-184, Jan. 2012.
30. SHERMAN J, MORRISON WJ. “Adjustment of a universe matrix corresponding to changes in a given column or row of the original matrix,” *Ann. Math. Stat.*, vol. 20, no. 4, pp. 621-622, Mar. 1949.
31. 曹海燕, 杨敬畏, 方昕等. 一种大规模MIMO系统低复杂度的CSM检测算法. 杭州电子科技大学学报(自然科学版), 2018, 38(1): 30-33.
32. Wei Chanjuan, Zhang Chunshui, Liu Jian, “Design of a fast matrix inversion method based on Cholesky decomposition,” *Electron. Des. Eng.*, vol. 22, no. 1, pp. 159-161, 2014.
33. Yin B, Wu M, Studerc E, et al. “Implementation trade-offs for linear detection in large-scale MIMO systems,” in *Proc. IEEE* *Int. Conf. Acoust., Speech Signal Process. (ICASSP),* Vancouver, BC, Canada, May. 2013, pp. 2679-2683.
34. 吕尉邦. 多用户MIMO系统中低复杂度预编码算法研究与实现. 上海: 上海交通大学, 2020.
35. E. Kanterakis, S. Moshavi, “Multistage linear receiver for DS-CDMA systems,” US. Patent 5757791. May.1998.
36. Kanterakis E, Moshavi S. Multistage linear receiver for DS-CDMA systems: US, US5757791[P]. 1998
37. Shen, Haipeng, and Jianhua Z. Huang. “Sparse principal component analysis via regularized low rank matrix approximation,” *J. Multivar. Anal.*, vol. 99, no. 6, pp. 1015-1034, Jun. 2008.
38. Witten, Daniela M., Robert Tibshirani, and Trevor Hastie. “A penalized matrix decomposition, with applications to sparse principal components and canonical correlation analysis,” *Biostatistics*, vol. 10, no. 3, pp. 515-534, Apr. 2009.
39. G. Golub, W. Kahan, “Calculating the singular values and pseudo-inverse of a matrix,” *J. Soc. Ind. & Appl. Math.* *Ser. B Numer. Anal.*, vol. 2, no. 2, pp. 205–224, Jan. 1965.
40. Schmidt, Drew. “A survey of singular value decomposition methods for distributed tall/skinny data,” in *Proc. IEEE/ACM 11th Workshop Latest Adv. Scalable Algorithms Large-Scale Syst. (ScalA)*, GA, USA, Sep. 2020, pp. 27-34.
41. Zhiqiang He, Yunhui Liu, and Xiaolin Zhang. 2006. “Performance of MIMO precoding systems with imperfect channel state information,” in *Proc. 3rd Int. Conf. Mobile Technol., Appl. & Syst. (Mobility)*, New York, NY, USA, Oct. 2006, pp. 66–es.
42. J. -B. Wang, M. Feng, X. Song and M. Chen, “Imperfect CSI based joint bit loading and power allocation for deadline constrained transmission,” *IEEE Commun. Lett.*, vol. 17, no. 5, pp. 826-829, May 2013.
43. 王景悟. 单双精度浮点运算加法器的实现. 西安: 西安石油大学, 2017.
44. J. H. Wilkinson, “Error analysis of direct methods of matrix inversion,” *J. ACM*, vol. 8, no. 3, pp. 281–330, Jul. 1961.
45. L. M. Yang, A. Fox, and G. Sanders. “Rounding error analysis of mixed precision block Householder QR algorithms.” *SIAM J. Sci. Comput.*, vol. 43, no. 3, pp. 1723–1753, May. 2021.
46. M. Connolly, “Probabilistic rounding error analysis for numerical linear algebra,” Ph.D. dissertation, University of Manchester, Manchester, UK, 2023.
47. O. Balabanov and L. Grigori. “Randomized Gram-Schmidt process with application to GMRES,” [Online]. Available: https://arxiv.org/abs/2011.05090.
48. J. Minango, A. Flores, P. Minango and D. A. Ibarra, “Low-complexity precoder for massive MIMO systems based on Gram-Schmidt conjugate direction method,” in *Proc. IEEE 3rd Ecuador Techn. Chapters Meeting (ETCM)*, Cuenca, Ecuador, Oct. 2018, pp. 1-5.
49. F. Dong, W. Wang and Z. Wei, “Low-complexity hybrid precoding for multi-user mmWave systems with low-resolution phase shifters,” *IEEE Trans. Veh. Technol*., vol. 68, no. 10, pp. 9774-9784, Oct. 2019.
50. Zhiqiang He, Yunhui Liu, and Xiaolin Zhang, “Performance of MIMO precoding systems with imperfect channel state information,” in *Proc. 3rd Int. Conf. Mobile Technol., Appl. & Syst. (Mobility)*, New York, NY, USA, Oct. 2006, pp. 66–es.
51. J. B. Wang, M. Feng, X. Song and M. Chen, “Imperfect CSI based joint bit loading and power allocation for deadline constrained transmission,” *IEEE Commun. Lett.*, vol. 17, no. 5, pp. 826-829, May. 2013.