电子电路的计算机辅助分析与设计方法

汪蕙　王志华　编著

清华大学出版社

(京)新登字 158 号

内 容 简 介

本书全面地介绍了电路 CAD 的基本理论和算法。全书共分 10 章,内容包括:电路方程的建立和求解,稀疏矩阵技术,元器件模型,线性和非线性电路直流、频域和时域的分析原理和计算方法,灵敏度计算,容差分析,最优化设计,以及目前大规模集成电路的一些新的分析方法。书中注重基本原理的论述,又反映了电路 CAD 领域的最新技术。书中所涉及的每一种分析方法都结合实际电路给出了相应的计算实例,并附有习题,便于读者阅读和理解。

本书适于作高等院校电类专业学生和研究生的教材,也可供从事电路设计和电路 CAD 软件开发的科技人员参考。

图书在版编目(CIP)数据

电子电路的计算机辅助分析与设计方法/汪蕙,王志华编著. —北京:清华大学出版社,1996

ISBN 7-302-02086-8

Ⅰ. 电… Ⅱ. ① 汪… ② 王… Ⅲ. ① 电子电路-计算机辅助电路分析②电路设计:计算机辅助设计 Ⅳ. TN710.9

中国版本图书馆 CIP 数据核字(96)第 02113 号

出版者:清华大学出版社(北京清华大学校内,邮编 100084)
印刷者:北京市清华园胶印厂
发行者:新华书店总店北京科技发行所
开 本:787× 1092 1/16 印张:18 字数:423 千字
版 次:1996 年 6 月第 1 版 1997 年 5 月第 2 次印刷
书 号:ISBN 7-302-02086-8/TN・70
印 数:5001—9000
定 价:15.00 元

前　言

电路CAD(Computer Aided Design)技术是模拟电路分析与设计的有力工具。随着集成电路的迅速发展,电子产品不断地推陈出新,以集成电路CAD为基础的电子设计自动化(Electronic Design Automation,简称EDA)已成为一个新兴的产业,渗入到集成电路设计的每一阶段。当前有近50%的集成电路设计是靠CAD和EDA工具完成的,而且这个比例还在不断地增长。为了帮助广大电路设计工作者尽快地掌握这门先进技术,指导电类专业的学生学习电路CAD方面的理论知识和设计工具,我们在多年教学和科研工作的基础之上编写了这本书。

全书共分10章。第1章介绍电子设计自动化和电路CAD技术的发展概况。第2章阐述了电路CAD的基本知识:建立电路方程的方法和求解方法。第3章以电路中有源器件为核心给出了二极管、BJT晶体管和MOS场效应管等半导体器件模型和一些常用的集成电路宏模型。第4章介绍线性和非线性直流分析,它是电路CAD中所有其它分析功能的基础。第5章论述瞬态分析方法。第6章介绍频域分析方法。第4章至第6章是本书的重点,这三章所介绍的直流分析、频域分析和瞬态分析,是电路CAD中应用最为广泛的三种分析方法。第7章讨论如何计算电路输出变量对电路元器件参数的灵敏度值。第8章阐述电路的容差分析算法和功能。第9章介绍目前求解大规模电路的一些新进展和算法。第10章简要地论述在电路CAD中常用的最优化算法。全书中各种分析方法都给出了计算实例,并附有设计工具的应用实例,便于读者理解和掌握。

编写此书过程中,刘润生教授自始至终给予了具体的指导,提出了许多极为有益的建议和详细的修改意见。本书原稿曾在清华大学电子工程系作为教材使用多年,多次改写过程中采纳了范崇治、王寒伟、杨华中、王和民等老师的许多建议,在此对他们的帮助表示感谢。谢源为附录中的C语言教学程序做了不少工作,也在此表示谢意。

限于水平,书中难免有错误及不当之处,望读者予以指正。

作者

1995年11月

目　　录

第1章 绪 论

随着集成电路与计算机的迅速发展,以电子计算机辅助设计(Computer Aided Design, CAD)为基础的电子设计自动化(Electronic Design Automation, EDA)技术已成为电子学领域的重要学科,并已形成一个独立的产业部门。它的兴起与发展,又促进了集成电路和电子系统的迅速发展。当前,集成电路的集成度越来越高,电子系统的复杂程度也日益增大,而电子产品在市场上所面临的竞争却日趋激烈,产品在社会上的收益寿命越来越短,甚至只有1到2年的时间。处于如此高速发展和激烈竞争的电子世界,电路设计工作者必须拥有强有力的EDA工具才能面对各种挑战,不断地创造出新的电子产品。

1.1 电子设计自动化的发展概况

电子设计自动化的发展大致可分为下述三个阶段。

70年代到80年代初期,电子计算机的运行速度、存储量和图形功能等方面还正在发展之中,电子CAD和EDA技术没有形成系统,仅是一些孤立的软件程序。这些软件在逻辑仿真、电路仿真和印刷电路板(PCB)、IC版图绘制等方面取代了设计人员靠手工进行繁琐计算、绘图和检验的方式,大大提高了集成电路和电子系统的设计效率和可靠性。但这些软件一般只有简单的人机交互能力,能处理的电路规模不是很大,计算和绘图的速度都受到限制。而且由于没有采用统一的数据库管理技术,程序之间的数据传输和交换也不方便。

80年代后期,是计算机与集成电路高速发展的时期,也是EDA技术真正迈向自动化并形成产业的时期。这一阶段,EDA的主要特点是:能够实现逻辑电路仿真、模拟电路仿真、集成电路的布局和布线、IC版图的参数提取与检验、印刷电路板的布图与检验、以及设计文档制作等各设计阶段的自动设计,并将这些工具集成为一个有机的EDA系统,在工作站或超级微机上运行。它具有直观、友好的图形界面,可以用电原理图的形式输入,以图形菜单的方式选择各种仿真工具和不同的模拟功能。每个工具都有自己的元件库,工具之间有统一的数据库进行数据存放、传输和管理,并有标准的CAM(Computer Aided Manufacture)输出接口。这种EDA系统能有效地完成自顶向下(top-down)的设计任务,即从电原理图构思到逻辑仿真、电路仿真、版图布局布线,一直到最后形成可以交付生产的IC版图的这一系列的自动化设计过程。这一时期比较成功的EDA系统有Mentor Graphics, Valid, Dazix等等。

进入90年代以后,EDA步入了一个崭新的时期。这个时期,微电子技术以惊人的速度发展,一个芯片上可以集成百万甚至千万个晶体管,工作速度可达到几个GB/s。电子系统朝着多功能、高速度、智能化的趋势发展,如数字声广播(DAB)与音响系统、高清晰度电视(HDTV)、多媒体信息处理与传播、光通信等电子系统。它们对集成电路和专用集成

电路(application specific IC, ASIC)的容量、速度、频带等都提出了更高的要求。这种高难度的集成电路要在短时间内正确地设计成功,就必须将 EDA 技术提高到一个更高的水平。另一方面,由于集成度的提高,上述的一个复杂电子系统可以在一个集成电路芯片上实现,这就要求 EDA 系统能够从电子系统的功能和行为描述开始,综合设计出逻辑电路,并自动地映射到可供生产的 IC 版图,我们称之为集成电路的高层次设计。因此 90 年代的 EDA 系统应具有如下特点:

(1) 真正具有自动化设计能力,能实现电路高层次的综合和优化。用户只要给出电路的性能指标要求,EDA 系统就能对电路结构和参数进行自动化的综合,寻找最佳设计方案,通过自动布局布线功能将电路直接形成集成电路的版图,并对版图的面积及电路延时特性作优化。目前数字电路的自动化综合与优化系统已相当成熟,典型代表是美国 Synopsys 公司的 EDA 综合系统。

(2) 具有开放式的设计环境。这种环境也称为框架结构(framework)。它在 EDA 系统中负责协调设计过程和管理设计数据,实现数据与工具的双向流动。它的优点是可以将不同公司的软件工具集成到一个统一的计算机平台上,使之成为一个完整的 EDA 系统,充分发挥每个设计工具的技术优势。设计者在这种设计环境中可以更有效地运用各种工具,提高设计质量和效率。

(3) 具有丰富的元器件模型库。EDA 系统需要各种不同层次、不同种类的元器件模型库的支持,或者说在电路设计的每个阶段,采用每个工具都要有不同的库支持。例如,原理图输入时需要元器件的外形库,逻辑仿真需要有逻辑单元的功能模型库,电路仿真需要模拟单元和器件的模型库,版图生成工具需要适应不同层次不同工艺要求的底层版图库等等。每一种库又按其层次分为不同层次的单元或元素库,例如逻辑仿真的库又以其行为级、寄存器级和门级分别设库。至于 VHDL 语言输入,所需的库更为庞大和齐全,几乎包括了上述所有库的内容。因此,各种模型库的规模与功能是衡量 EDA 工具优劣的一个重要标志。

综上所述,一个 EDA 系统的组成应该是:

框架结构 (framework) + 各种 CAD 工具 (tools) + 模 型 库 (libraries) + 算 法 (methodology) + 硬件支撑环境 (support)

目前,在国际 EDA 系统排行榜中,列于首位的几家 EDA 公司是:Mentor Graphics, Cadence, Synopsys 和 Viewlogic 公司。

1.2 EDA 系统的设计工具

一个能够完成较为复杂的 VLSI 设计的 EDA 系统至少应包括 10～20 个 CAD 工具,图 1.2.1 描绘了一个典型的自顶向下设计的 EDA 系统框图,下面我们对这个框图中所涉及的主要 CAD 工具进行简单介绍。

1. 逻辑综合与优化

逻辑综合是 90 年代电子学领域兴起的一种新的设计方法,是以系统级设计为核心的高层次设计工具,也称之为概念驱动工程。它把最新算法与工程界多年积累的设计经验结

图 1.2.1　EDA 系统框图

合起来,自动地将采用 VHDL 硬件描述语言描述的电子系统综合成为满足设计性能指标要求的逻辑电路,并对电路进行速度、面积等方面的优化,最后形成所要求工艺条件下的集成电路版图。VHDL(VHSIC hardware description language)的全称是超高速集成电路硬件描述语言,简称硬件描述语言。它是 EDA 系统的一种输入模式,支持从数字系统级到门级的多层次的硬件描述,与具体的技术和工艺无关,能够抽象出数字硬件设备的本质特性,即行为、时间关系和结构特性。这些特性使 VHDL 能在多种场合对硬件设备进行统一描述,因此,用 VHDL 所实现的仿真无论在计算速度还是在容量上都远远超过了传统的描述方式。

高层次的逻辑综合一般包括 VHDL 语言形式的设计输入,行为级至门级的仿真和综合优化,版图验证与后仿真以及测试综合。

(1) VHDL 设计输入

VHDL 硬件描述语言的输入方式是 EDA 系统的主要输入方式。统计资料表明,在 VHDL 和原理图两种输入方式中,前者约占 70% 以上,这个趋势还在继续增长。其主要原因是 VHDL 支持多种不同设计方法,而与具体技术和工艺无关。VHDL 能够实现从数字系统级行为到门级硬件的描述,并能在多种描述层次上立即进行操作,应用这些描述层次的任一组合来模拟系统,因此功能强,应用面广。例如,我们可以采用 VHDL 定义一个系统的顶层行为和结构以及它所包含的各种模块,这些模块可以是控制单元、算术逻辑单元(ALU)、RAM、ROM 或数据通道等等,VHDL 的编译器会自动将这些顶层模块描述转换成底层的功能描述。应该指出,VHDL 尚有语法繁琐、用户学习周期长、难于掌握的缺点,所以目前图形输入方式又有新的进展,出现了一种高层次的图形控制设计方法。它可以对数字系统的状态流程图、进程图、数据流程图和原理框图进行处理,将它们转化为 VHDL 文件。这为高层次的综合仿真提供了更大的方便和灵活性。

(2) 电路的综合优化

综合一词来源于英文单词 synthesis, 其含义是把一些抽象的对象组合为一个统一的整体。在集成电路设计中, 综合则意味着将集成电路设计中设计描述的一种形式自动地向另一种形式变换的过程。因此, 数字电路的综合是将行为级的系统描述转化为门级描述, 同时进行门级电路的优化; 甚至可以进一步将门级电路映射到某种工艺的版图, 并对版图进行优化。这后一步, 有时也称为版图综合。电路优化是在各种设计要求的约束下进行电路性能、速度方面的优化; 版图优化是在指定工艺条件下对版图面积进行优化。这两种优化过程是互相关联的, 需要通过版图的参数提取, 得到版图对电路参数影响的信息, 再对电路进行反复验证(通常称为后仿真), 最后达到电路速度和面积的整体优化。

(3) 测试综合

测试综合包括两方面: 可测性设计和自动测试矢量生成。也可以将测试综合看作是逻辑综合的一个组成部分。根据设计要求, 在逻辑综合中必须考虑可测性因素, 生成相应故障覆盖率的可测性电路, 并生成测试矢量。它还可以在已设计好的电路中插入测试电路结构, 以满足故障覆盖率的要求。

2. 混合设计方法

随着集成电路复杂程度的不断提高, 各种不同学科技术、不同模式、不同层次的混合设计方法已被认为是 EDA 系统所必须支持的方法。

(1) 不同学科技术的混合方法(mixed-discipline), 指电子技术与各种非电学科技术的混合设计方法。例如: 电-机械, 电-化学, 电-光学, 电-应力, 电-磁, 电-热学等方面的混合设计方法。

(2) 不同模式的混合方法(mixed-mode), 一般是指数字电路与模拟电路的混合, 数字电路、模拟电路与数字信号处理(DSP)技术的混合, 电路级与器件级的混合方法等等。

(3) 不同层次的混合方法(multi-level), 指逻辑设计中行为级、寄存器级、门级, 以及开关级的混合设计方法。

目前在各种应用领域, 如数字电路、模拟电路、DSP 的专用集成电路, 多芯片模块(multi-chip module, 简称 MCM)以及印刷电路板系统的设计都需要采用各种混合设计方法。

3. 集成电路版图设计方法

在自顶向下的设计过程中, 完成电路综合优化或电路仿真之后, 就进入 IC 芯片的版图设计阶段。

集成电路的版图设计方法大致可归纳为四种。

(1) 全定制设计方法(full-custom)

全定制设计方法一般是为满足一些特殊的要求, 而又没有现成的单元库可以借鉴的情况下采用的设计方法。这些要求可以是高速度、低功耗、最小面积, 也可能是一些特殊结构或性能等等。它需在应用版图设计工具进行设计过程中不断地插入人工干预, 以便把每一个器件位置、连接关系都安排得最紧凑、最恰当, 因此是一种特别花费人力和时间, 同时也是性能最佳的一种设计方法。对于那些生产批量很大, 使用生命力较强的计算机、通信和声象等系统的 IC 产品, 通常采用全定制设计方法来实现。

(2) 标准单元法(standard cell)

这种设计方法是以各IC生产厂家的单元库为基础,根据电路要求从库中调取所需要的单元,进行自动布局布线,生成掩膜版图。它的目标是在满足工艺约束和电性能条件下,使版图的面积最小。目前先进的布局布线工具可以支持多层布线,能对版图面积进行压缩优化并保证设计规则。允许各种高度的标准单元和各种宽度的芯片管脚,布局布线速度十分迅速,并保证100%的布通率和最小芯片面积。

(3) 门阵列法(gate array)

门阵列和门海(sea of gate)是预先在IC芯片上生成由基本门或单元组成的阵列,即完成了除连线以外的所有芯片的加工步骤。再根据用户的电路要求,选择预制芯片阵列中适当的单元进行布局布线,最后只需给出金属连线和通孔掩膜版图。这种设计方法的最大特点是设计周期短,设计和制造成本都较低。缺点是阵列中单元利用率低,芯片面积也较大。

(4) 可编程逻辑器件方法(programmble logic device)

可编程逻辑器件是一种通用芯片,由逻辑单元阵列组成,特点是"可编程",即可以通过编程去选择逻辑单元和互连关系,将阵列安排成适合于逻辑功能的专用芯片。这些逻辑单元可以是门、触发器或较大的宏单元。逻辑单元之间的连线是可编程的开关系统,一般采用EPROM或E^2PROM作可编程器件,通过它们将布局布线结果映射到逻辑阵列中,形成芯片。例如较早打入中国市场的美国Xilinx公司的FPGA可编程芯片,是采用先进的CMOS工艺制造的,它的基本结构是由可编程逻辑模块(CLB)、输入输出模块(IOB)和可编程连线组成。每个CLB中包含组合逻辑电路、D触发器和内部控制电路,通过查表方式实现布尔逻辑功能。它的可编程连线资源是由两层栅格状金属线段构成。两层金属线段的交叉点上置有专门的连通晶体管,每个晶体管由一位组合比特控制通断,从而形成金属线间的可编程连接点。此外,这种晶体管还用于构成开关阵列,通过编程实现所选金属线段和CLB,IOB引脚之间的连通。

这种设计方法与前面所述的版图设计方法不同,它所实现的不是待生产的IC版图,而是可以实际应用的芯片。这种可编程逻辑器件的出现,使设计工程师在实验室里就可以设计出自己的ASIC器件。其中以近几年发展起来的EPLD(可擦除的可编程逻辑器件)和FPGA(现场可编程门阵列)最受重视。用户只要购置了EPLD或FPGA的芯片,并有计算机和相应的软件开发系统,就可以自己动手设计芯片。其设计步骤是:将电路以原理图或VHDL形式输入,然后进行逻辑仿真,保证逻辑功能正确,接着对芯片中的逻辑单元进行布局布线,并对布通的电路进行后仿真,最后通过编程器和烧结器将其制成实用的ASIC芯片。这个过程一般只需几个小时。此外,还有一个突出的特点是它的可擦除性。由于采用EPROM和E^2PROM做为互连开关,可以通过紫外线或电实现擦除,即反复编程,也就是说一个FPGA(或EPLD)芯片可以反复设计和使用。这样,不仅大大地缩短了设计周期,减少了投片带来的高昂费用,而且降低了设计风险。对于在实验室中进行科研项目所实现的ASIC及一些批量不大的ASIC开发产品,采用EPLD和FPGA的形式设计芯片,在设计成本、时间上都是十分合算的。所以,目前越来越多的科研部门和高等院校在科研工作中采用可编程逻辑器件来实现ASIC,这种方法已成为设计电子系统的一个

非常重要的手段。

当前，Xilinx，AT&T，Altera，Lattice 等公司生产的 FPGA 和 EPLD 芯片及开发系统在国内应用较为广泛，从几千门到几万门的 FPGA(EPLD)，一般价格为十几到几百美元。

4. 集成电路版图的电参数提取与校验

集成电路的版图需要通过校验才能确保其正确和可靠。这种校验一般包括：几何设计规则检查、电规则检查、版图与电路图一致性检查以及版图的电参数提取。

采用自顶向下的设计方法，通过自动布局布线形成的版图，由于自动化程度高，中间没有人工干预，一般不会产生设计过程中的错误，故可以不必进行前三项检验。但版图的电参数提取以及将提取出的电参数加到原始的原理图中再进行后仿真，则是一项十分重要的检验工作。由于版图基本上是由各种不同掩膜层图形组成的，对版图进行布局布线优化时是以版图面积最小和 100% 的布通率为目标，但这两项指标是矛盾的，常常采取折衷方案，因此无论版图中单元面积大或小，单元间连线长或短都会产生不同数量级的寄生电容、电感和电阻，这是原电路设计中所没有的，必须考虑这些寄生参数对电路特性的影响，以保证电路特性的正确。版图参数提取工具从版图的几何图形中识别和提取这些寄生参数，并通过再一次仿真检验这些寄生参数对电路的影响。如果电路性能有变化，如速度、延迟等指标不满足，则需进一步改善版图设计，以确保集成电路特性的正确性。

5. 友好的设计环境

一个优秀的 EDA 设计系统必须有良好的设计环境。它包含两个方面。

(1) 友好的用户界面

友好的用户界面使用户能在灵活、易用、图文并茂的环境中应用各种不同层次的软件工具。它的主要特点是：

1) 图形和 VHDL 形式的输入；

2) 多重观察窗口，可以同时观测同一电路在不同层次下的设计情况，直观、便于比较；

3) 键盘、菜单、功能键和手绘符号等多种命令选择方式；

4) 元器件库和相应图表的浏览、管理；

5) 输出波形的处理，既可以观察仿真现场波形，也可以对已模拟出的波形做各种综合处理。

(2) 框架结构

框架结构是 EDA 的操作系统，主要完成数据库、数据处理和网络通信的功能。它的主要特点是：

1) 开放性和集成化。支持统一的服务平台和独立的编辑接口，将各种软件工具很方便地放置在一起，并提供软件工具、硬件和网络之间的协调，安排设计流程和策略，创造统一的设计环境。

2) 合理的组织。框架结构简化了工具的使用，实现了数据跟踪和数据在工具间的双向流动。允许设计项目在前台现场观察和后台处理数据同步进行，并能及时了解任何一个环节的数据变化对整体的影响。

3) 易于使用。它给用户提供了基本的工业标准 X-Window 和灵活的用户接口，使用户能极为方便地应用 EAD 系统。

框架结构的出现，使国际上许多优秀的 EDA 工具合并在一个统一平台上的愿望得以实现。美国几家著名的 EDA 公司 Cadence，Mentor Graphics 等都相继采取了合并措施，利用框架结构集各家精华于一体，使 EDA 系统所涉及的领域更加宽广，也为其应用开辟了更加广阔的前景。

1.3 模拟电路 CAD 的发展概况

1.3.1 模拟集成电路的特点和设计自动化方法

模拟集成电路的设计与数字电路设计有很大的区别。数字电路可以很方便地抽象出逻辑门、寄存器、加法器、减法器等不同层次的逻辑单元，还可以用数据流图、有限状态模型等形式进行高层次描述，并将这些逻辑单元和高层次行为描述用于不同层次的电路设计。数字电路这种结构简单、规则、易于抽象化的特点极大地促进了数字电路设计的自动化。而模拟集成电路则要复杂得多。首先，模拟电路种类繁多，性能通常与连续变化的变量有关。比如：放大器的幅度与增益不仅仅是电路拓扑结构的函数，而且与元件参数和工作频率等都有关系。其次，模拟电路的结构也千差万别，同样的电路特性可由不同的电路结构实现，不同参数的同一电路结构会得到不同的电路性能。还有，模拟电路与工艺条件、工作环境等直接相关，同一芯片上的不同电路之间干扰也相对比较大。模拟电路的这些特点使其性能和结构的抽象提取和表述都较为困难，从而也就给模拟电路的层次化设计，尤其是高层次设计带来了较大困难。从前面介绍的 EDA 系统也可以看出，集成电路的自动化设计技术始终偏重于数字电路设计，对于数字电路，从高层次的自动综合到最低层次的版图设计的布局布线都有成熟的实用软件工具，而且能实现从顶到底的整体自动设计。而模拟集成电路的自动设计技术远没有数字电路的相应技术成熟，还不能实现完善的高层次仿真，更不用说自动综合了。但从应用角度而言，目前对模拟集成电路需求量也在逐年增加，尤其是在卫星通信、导航、传输等领域，在高频、低噪声、大功率及并行处理的应用条件下，模拟电路比数字电路更有优势。另外，由于自然界的绝大多数信号如语音、图象等信号皆为模拟信号，既便要作数字化处理和传输，模拟电路也是必不可少的。但是，由于模拟集成电路设计工具的局限性，模拟集成电路和数字/模拟混合集成电路设计的难点就主要在模拟电路部分。在数模混合集成电路中，模拟电路部分所占的比例往往不大，但设计时间却可能很长。图 1.3.1 中给出了数模混合集成电路中，数字与模拟部分所占比例和它们在设计时间上的相对关系。基于上述原因，工业界在集成电子系统设计中急需有效的模拟电路和数模混合电路设计的 CAD 工具，并迫切地要求模拟及数模混合集成电路设计的自动化。

1.3.2 模拟集成电路的设计工具

目前国际上各大 EDA 公司的模拟电路设计工具，都是以美国加利福尼亚大学柏克莱(Berkeley)分校开发的 SPICE 程序为基础实现的。SPICE 程序是一个用于模拟集成电

图 1.3.1　数模混合设计中电路规模与设计时间关系示意图

路的电路分析程序, 首次于 1972 年推出, 是用 FORTRAN 语言写的, 1975 年推出正式实用化版本。自 SPICE 问世以来, 其版本不断更新, 其中以 1981 年的 SPICE 2G.5 版本最为流行, 是 80 年代世界上应用最广的电路设计工具, 1988 年 SPICE 被定为美国国家工业标准。1985 年, Berkeley 将 SPICE 重新用 C 语言改写, 称为 SPICE3, SPICE3 采用 C 语言后在数据结构和执行效率上都有很大改善。多年来, SPICE 版本更新主要体现在: 电路分析功能的扩充, 算法的改进与完善, 元器件模型的增加, 以及输入、输出界面的改善。各种以 SPICE 为核心的商用电路仿真工具, 在 SPICE 的基础上做了许多实用化工作, 例如输入是以原理图方式, 输出绘图处理更加灵活, 求解收敛性更好, 模拟功能有所扩充, 增加了宏模型, 以及备有较为丰富的元器件模型参数库等等。一些在国际上享有盛誉的 EDA 系统中都有相当强的模拟电路仿真工具, 它们是 Mentor Graphics 公司的 AccuSim, Cadence 公司的 Cadence SPICE, Analogy 公司的 Saber, 以及 Meta Software 公司的 HSPICE, Micro Sim 公司的 PSpice 等等, 这些工具在国内也颇为流行。下面我们就 Berkeley 的 SPICE3, 以及各种商用电路仿真工具中所能实现的分析功能和元器件模型种类做简单介绍。

1. SPICE 的仿真功能

(1) 直流分析

1) 计算电路的直流工作点, 即在电路中电感短路、电容开路的情况下, 求得电路的每一个节点电压和电源支路电流。

2) 计算电路的直流小信号传输函数, 即计算在直流小信号工作下的输出变量和输入变量之比值。

3) 计算直流转移特性曲线, 即在用户指定范围内计算电路输出变量与指定的输入源步进变化之间的关系曲线。

同时, 直流分析是瞬态分析和交流小信号分析的基础, 通常用直流分析决定瞬态的初始条件和交流小信号分析时电路中非线性器件的小信号模型参数。

(2) 交流小信号分析

交流小信号分析是在正弦小信号工作条件下的一种频域分析, 是一种线性分析方法。

1) 频域分析, 即计算电路的幅频特性和相频特性。

2) 失真分析, 是假定输入端加有一个或两个信号频率时, 在输出负载上计算出的三次以下的小信号谐波失真功率。

3) 噪声分析,即计算每个频率点上指定输出端的等效输出噪声和指定输入端的等效输入噪声电平。

(3) 瞬态分析

瞬态分析是一种非线性时域分析方法。

1) 在用户指定的时间区间内,进行电路的瞬态特性分析。

2) 在大信号正弦激励情况下,对输出波形进行傅里叶分析,计算出基波和 2～9 次谐波系数,以及失真系数。

(4) 灵敏度分析

灵敏度分析是计算电路指定的输出变量对电路元器件参数的小信号灵敏度值。

1) 直流灵敏度分析,是指电路直流分析时,计算出指定的输出变量对电路中所有元件参数和晶体管的所有模型参数单独变化的灵敏度值,包括绝对灵敏和相对灵敏度值。

2) 交流小信号灵敏度分析,是在指定频率范围内每个频率点上计算电路输出变量对电路全部元器件参数的灵敏度值。从 SPICE3F 版本中,才开始有交流小信号灵敏度分析能力。

(5) 零极点分析

零极点分析是计算电路的传输函数的零点和极点。通过求得的零极点分布,用户可以确定电路的频响特性、时域特性,以及判断电路的稳定特性。

(6) 容差分析

容差分析是计算电路中元器件参数偏离标称值情况下,对电路输出特性的影响。

1) 蒙特卡罗(Monte-Carlo)分析,是一种统计分析方法,在给定电路中元器件参数容差的情况下,计算电路输出变量的均值和标准偏差。如果同时指定电路输出变量的容差,还可以计算出电路输出特性的合格率及合格率的偏差。

2) 最坏情况(Worst-Case)分析,是指电路中所有元器件参数都处于其容差边界的一种最坏组合情况下,计算出电路输出特性最大偏差的上界和下界值。

大多数商用电路仿真软件中都有蒙特卡罗分析和最坏情况分析功能。

(7) 温度特性分析

通常 SPICE 程序是在标称温度(27℃或 300K)情况进行各种分析和仿真的。如果用户指定电路的工作温度,则 SPICE 可以进行不同温度下的电路特性分析。

(8) 优化设计

电路的优化设计是在给定电路拓扑结构和电路性能约束的情况下,确定电路元器件的最佳参数组合。目前只有少数商用电路仿真软件具有优化能力,例如美国 Meta Software 公司的 HSPICE,具有直流、交流小信号和瞬态优化功能。

2. SPICE 的元器件模型

(1) 一般元件。指电阻,线性和非线性电容,线性和非线性电感,互感,三种形式传输线:有损和无损传输线、均匀分布 RC 传输线,以及电压和电流控制开关。

(2) 半导体器件。指结型二极管,双极型晶体管,结型场效应管,六种不同级别的 MOS 场效应管:MOS1～3、BSIM1～2 和 MOS6,砷化镓器件。

(3) 电源、信号源。电源指独立电流源,独立电压源,四种线性和非线性受控源:

VCCS, CCCS, VCVS, CCVS。信号源有: 脉冲源、正弦源、指数源、分段线性源、单频调频源等等。

新版本的 SPICE3 和商用电路仿真工具中还增加了表格形式源、数学函数源、拉氏变换源等, 这些源也可以看做是一种数学描述的宏模型。

(4) 宏模型。指运算放大器、电压电流比较器、开关电源变换器、可控硅器件、相乘器、石英晶体振荡器、磁芯等等。

大多数商用电路仿真软件工具中都建有元器件模型参数库。除了通用的模型库以外, 还有按照晶体管和 IC 芯片的种类和生产厂家分别建立的专用晶体管参数库和 IC 宏模型参数库。用户可以很方便地查询和调用这些库, 给仿真带来极大的便利。

3. 模拟电路设计工具的基本内容和组成

模拟电路设计工具的基本组成和仿真步骤如图 1.3.2 所示。

图 1.3.2　模拟电路仿真程序框图

(1) 人机交互界面

人机交互界面是计算机、软件程序与用户之间的桥梁。为了用户能方便地使用 CAD 系统, 简洁、直观、易于掌握的友好的人机交互界面是十分必要的。人机交互界面由以下几部分组成:

1) 菜单形式的命令集。用户在使用 CAD 系统时所需用的操作命令, 如选择、调用、转换和运行等等, 都可以通过图形菜单或弹出式菜单、下拉式菜单来提示用户实现操作。由于菜单通过图形或文字提示出用户应选择的命令, 无需用户学习和记忆大量繁琐的命令语言, 既简单明了, 又直观、易于掌握, 给用户使用 CAD 系统带来很大方便。

2) 交互式图形输入。人机交互界面充分发挥计算机显示终端的图形功能, 采用形象、直观的图形和文字式的对话方式, 用户可以用鼠标器、光笔等输入设备, 直接在屏幕上绘制电原理图。

3) 多窗口、多进程作业方式。利用 CAD 工作站的多窗口管理系统, 用户可以同时在显示屏幕上开若干个窗口, 进行不同的作业。例如, 一个窗口绘制电原理图, 另一个窗口进行模拟设计, 再一个窗口可以实时地绘制仿真结果波形, 极大地提高了设计效率。

(2) 输入方式

模拟电路的仿真输入基本上是以原理图和网单文件两种形式。以原理图形式输入比较简单、直观。从电路符号图形库中调出所需电路符号, 组成电路图, 由原理图编译器自动

将原理图转化为电路网单文件,并自动标上节点号,提供给仿真工具进行仿真。如果用户熟悉仿真输入语言,又不必有原理图存档,也可以直接输入电路网单文件。

随着模拟电路仿真工具中宏模型在数量、种类、规模上的增加,模拟电路的硬件描述语言(AHDL 或 HDLA)已经问世,并且也即将成为工业标准,AHDL 也将成为模拟仿真的一种有效的输入形式。

(3) 元器件模型的建立与处理

进行电路仿真需要构造电路元器件的模型,即用数学模型来代替具体的物理器件,这种数学模型应能正确地反映器件的物理特性和电学特性,并便于在计算机上做数值计算。建模工作在电路 CAD 中起着举足轻重的作用,它直接影响着整个仿真的精度和速度。SPICE 程序近年来的发展也主要是在建模方面,例如随着集成电路集成度的提高,MOS 场效应管模型的沟道长度越来越短,因此需要用高阶效应模型来描述,于是 SPICE 中有 MOS1～MOS6 不同精度的模型。但也不是模型精度越高越好,因一般而言,精度越高也就越复杂,需要越多的计算时间和存储空间。所以,在电路 CAD 中是根据应用条件和分析目的来确定模型种类。一般同一个器件,针对不同的分析要采用不同的模型,如直流非线性模型、交流小信号模型、大信号瞬态模型等等。在确保模拟正确的条件下,尽可能采用简单的模型。目前,随着 VLSI 的飞速发展,电路的规模越来越大,仅用元件级模型来构成电路,其电路方程将十分庞大,使计算机在时间和容量上都难于接受。因此,建立电路的宏模型,是目前建模工作的一个发展趋势。

在电路仿真过程中,相当多数的时间是用在处理器件模型的数学方程上,即根据用户所输入或选用的器件类型,按照模型参数计算出电路方程等效导纳、电流源等信息。由于器件方程大多是非线性方程,故采用各种算法进行处理时均较复杂,需花费较多的时间。

(4) 电路方程的建立与求解

电路仿真中求解电路方程多采用数值方法。对不同分析领域,电路方程的形式不同,求解方法也不同。例如,交流小信号分析,所列方程是线性代数方程,可采用高斯消元法或 LU 分解法求解;直流非线性分析,所列方程是非线性代数方程,通常采用牛顿-拉夫森方法迭代求解;瞬态分析,所列方程是常微分方程,一般用变步长隐式积分法求解。另外,电路方程还有一些本身固有的特点,例如电路中元器件的参数值可能相差很大(可达 10^6 倍以上),因而求解线性代数方程的数值稳定性问题,及解微分方程的稳定性问题都要特别注意和处理。再则,由于电路中半导体器件的非线性特性,解非线性方程时遇到的不收敛问题,也是电路 CAD 算法中的一大难题。最后还应说明,电路方程通常是稀疏的,因而求解线性方程组的稀疏矩阵技术得到广泛的应用。近年来,在求解大型的,特别是数字电路方程方面,解方程的算法又有新的进展,各种松弛算法、撕裂算法及休眠技术等对于提高算题速度和效率十分有效。总之,研究解线性、非线性方程以及微分方程的算法,对提高电路 CAD 分析精度、加速分析速度、减少计算机存储容量等有很大关系。

(5) 绘图处理

电路仿真工具大多有一个独立的输出绘图处理软件包,它的作用是将电路模拟结果绘制成标准、直观的波形或曲线,便于观察、输出或存档。绘图可以是实时的,即边计算边绘出波形;也可以在仿真完成以后统一做绘图处理,例如对不同节点的输出波形做算术运算,或对不同电路的输出结果在同一画面上进行比较。

第 2 章　电路方程的建立和求解方法

电路分析和设计的首要步骤就是将一个实际的电路用一整套数学方程来描述,因而电路 CAD 首先要解决如何用计算机自动地建立电路方程和求解电路方程。

利用计算机自动建立电路方程的方法很多,有节点法、改进节点法、表矩阵法和双图法等等。这些方法在建立电路方程时所选择的变量性质和数量不同,因而方程的形式和数目也不相同。但是电路方程的建立都是从原始数据出发,以网络拓扑方程和元件支路特性方程为基础,经过方程变换而实现的。电路方程都以矩阵形式表达,清晰直观,易于在计算机中进行计算。在电子电路的计算机辅助分析中,目前使用最广泛也是最简便的方法是节点法和改进节点法,它们主要以节点电位为变量。因电路中的节点数一般远小于支路数或回路数,因此用节点法或改进节点法所列方程组的独立方程数也较少,建立节点方程方法也较为简单。在这一章,我们用较多的篇幅介绍节点法和改进节点法。

建立方程以后,下一步要解决的就是如何求解方程。本章主要介绍线性代数方程组的求解问题。线性代数方程组的求解本身是个纯数学问题,而且是个非常经典和相当成熟的问题,在线性代数方面的计算方法中已有了详细的研究与介绍,而且有不少标准算法程序可供参考。因此在这里我们就不对算法做更详细的说明,只简单地介绍电路 CAD 中常用的两种方法:高斯消去法和 LU 分解法。

2.1　建立电路方程的常用方法

电网络方程一般以节点电位、支路电压或支路电流作为变量,电网络方程必须满足两类基本的约束条件:

(1) 电路的支路电压和电流必须分别遵循基尔霍夫电流定律 KCL(Kirchhoff current law)和基尔霍夫电压定律 KVL(Kirchhoff voltage law)的约束关系。

(2) 电路的支路电流和支路电压之间必须遵循 VCR(voltage current relationships)定律(即电压电流关系)的约束关系。

无论是频域分析还是时域分析,稳态分析还是瞬态分析,所列的方程都应遵循这两类约束。在第一类约束条件中,KCL 定律反映各支路电流之间的关系,KVL 定律反映各支路电压之间的关系。这些关系是由电路的拓扑形式决定的,与支路特性无关。在电路理论中,我们常用网络拓扑矩阵来表示 KCL 方程和 KVL 方程,即有

$$\text{KCL:} \qquad AI_b = 0 \tag{2.1.1}$$

$$\text{KVL:} \qquad A^T V_n = U_b \tag{2.1.2}$$

其中,A 为基本关联矩阵,它反映电网络中节点与支路之间的关系。I_b 为支路电流向量,U_b 为支路电压向量,V_n 为节点电位向量。在下面介绍的列方程的各种方法中,我们会经常引用这两个方程。

2.1.1 表矩阵法

KCL 方程、KVL 方程和 VCR 支路特性方程是描述电路拓扑及特性的三个基本方程。表矩阵法就是将这三个基本方程组合在一起,形成一个大型的矩阵方程。它把支路电压、支路电流和节点电位都作为方程的变量,因而所形成的方程组阶数较高。其系数矩阵中非零元素稀少,故也称为稀疏表格法。这种列方程方法比较直观,步骤简单,易于实现。

表矩阵法列方程的步骤如下:

(1) 由电路的拓扑信息建立其关联矩阵 A。

(2) 建立由关联矩阵 A 表示的 KCL 方程和 KVL 方程。

KCL 方程:

$$AI_b = 0$$

KVL 方程:

$$A^T V_n - U_b = 0$$

其中,I_b 是支路电流向量,U_b 是支路电压向量,V_n 是节点电位向量。

(3) 建立各支路特性方程。

支路特性方程的一般形式为:

$$\begin{matrix} Y_b \\ K_2 \end{matrix} U_b + \begin{matrix} K_1 \\ Z_b \end{matrix} I_b = \begin{matrix} W_{b1} \\ W_{b2} \end{matrix} \tag{2.1.3}$$

其中,Y_b 为支路导纳矩阵,Z_b 为支路阻抗矩阵,K_1 和 K_2 为无量纲的常数,W_{b1}和 W_{b2}分别表示独立电流源、独立电压源以及初始条件对电容和电感的影响。为了使表达式更加紧凑,可以采用下面的简化形式:

$$Y_b U_b + Z_b I_b = W_b \tag{2.1.4}$$

我们将电路中常用的二端元件在式(2.1.4)中所对应的 Y_b, Z_b 和 W_b 的各元素值列于表 2.1.1 中。

表 2.1.1 支路特性方程表

元件	特性方程 $Y_bU_b+ Z_bI_b= 0$	Y_b 值	Z_b 值	W_b 值
电阻	$U_b- R_bI_b= 0$	1	$- R_b$	0
电导	$G_bU_b- I_b= 0$	G_b	- 1	0
电容	$SC_bU_b- I_b= C_bU_0$	SC_b	- 1	C_bU_0
电感	$U_b- SL_bI_b= - L_bI_0$	1	$- SL_b$	$- L_bI_0$
电压源	$U_b= E$	1	0	E
电流源	$I_b= I_S$	0	1	I_S

(4) 将(2.1.1)、(2.1.2)和(2.1.4)式合并起来,即得到

$$U_b - A^T V_n = 0$$

$$Y_b U_b + Z_b I_b = W_b$$

$$A I_b = 0$$

写成矩阵形式:

$$\begin{bmatrix} I & 0 & -A^T \\ Y_b & Z_b & 0 \\ 0 & A & 0 \end{bmatrix} \begin{bmatrix} U_b \\ I_b \\ V_n \end{bmatrix} = \begin{bmatrix} 0 \\ W_b \\ 0 \end{bmatrix} \tag{2.1.5}$$

这就是用表矩阵法列出的表矩阵方程组,它是非线性代数方程和非线性常微分方程的混合方程组。系数矩阵中各子矩阵均为方阵。对于电容和电感支路,算子 S 在频域分析中为 $j\omega$,在时域分析中为 d/dt。

如果电路网络中有 n 个独立节点,m 条支路,则表矩阵方程(2.1.5)是个(2m+ n)×(2m+ n)维的方程组。对多数电路,其支路数往往都大于甚至远大于节点数,因此表矩阵方程组的维数是很高的。从式(2.1.5)中还可以看出,表矩阵方程组的系数矩阵中除了 4 个零元素的子阵外,Y_b, Z_b 和单位阵 I 都是大多数元素为零元素的子阵,因此系数矩阵是个十分稀疏的矩阵。

例 2.1.1　某电路及拓扑图如图 2.1.1(a)、(b)所示。列出该电路的表矩阵方程。

图 2.1.1　例 2.1.1 电路及拓扑图

首先,由电路拓扑图建立关联矩阵 A:

$$A = \begin{bmatrix} -1 & 1 & 1 & 0 \\ 0 & 0 & -1 & 1 \end{bmatrix}$$

根据元件支路特性方程建立的 Y_b 子阵和 Z_b 子阵为

$$Y_b = \begin{bmatrix} 0 & 0 & 1 \\ 1 & & \\ & SC_3 & \\ 0 & G_4 & \end{bmatrix} \qquad Z_b = \begin{bmatrix} & & -R_2 \\ & & \\ & & -1 \\ & & -1 \end{bmatrix}$$

则表矩阵方程为

$$\begin{bmatrix} 1 & 0 & -A^T \\ Y_b & Z_b & 0 \\ 0 & A & 0 \end{bmatrix} \begin{bmatrix} U_b \\ I_b \\ V_n \end{bmatrix}$$

$$
\begin{array}{c|c|c|c}
\begin{matrix} 1 & & & \\ & 1 & & \\ & & 1 & \\ & & & 1 \end{matrix} & & \begin{matrix} 1 & 0 \\ -1 & 0 \\ -1 & 1 \\ 0 & -1 \end{matrix} & \begin{matrix} U_{b1} \\ U_{b2} \\ U_{b3} \\ U_{b4} \end{matrix} \\
\hline
\begin{matrix} 0 & & & \\ 1 & & & \\ & SC_3 & & \\ & & G_4 & \end{matrix} & \begin{matrix} 1 & & & \\ & -R_2 & & \\ & & -1 & \\ & & & -1 \end{matrix} & & \begin{matrix} I_{b1} \\ I_{b2} \\ I_{b3} \\ I_{b4} \end{matrix} \\
\hline
 & \begin{matrix} -1 & 1 & 1 & 0 \\ 0 & 0 & -1 & 1 \end{matrix} & & \begin{matrix} V_1 \\ V_2 \end{matrix}
\end{array}
=
\begin{array}{c}
\begin{matrix} I_S \\ 0 \\ 0 \\ 0 \end{matrix} \\
\hline
\\
\hline
\begin{matrix} 0 \\ 0 \end{matrix}
\end{array}
$$

含有受控源的电路也能用表矩阵法列出电路方程。四种常用的受控源是：电压控制电流源(VCCS)，电压控制电压源(VCVS)，电流控制电流源(CCCS)，电流控制电压源(CCVS)。它们是二端口器件，对每一端口用一支路特性方程描述，即对每一种受控源都由两个支路方程描述其特性。在表 2.1.2 中列出了四种受控源的符号及支路特性方程。

由上所述，表矩阵法列方程比较容易，方程比较稀疏，并可以直接求解出电路中任何一个变量。但由于方程阶数较高，且矩阵的结构不对称，处理程序较为复杂，存储量和计算量也较大。美国著名的电路仿真程序 ASTAP 采用此法建立方程。

表 2.1.2　二端口器件支路方程

元　　件	符　　号	支路特性方程 $Y_bU_b+Z_bI_b=0$
VCCS		$\begin{matrix}0 & 0\\ g_m & 0\end{matrix}\begin{matrix}U_1\\U_2\end{matrix}+\begin{matrix}1 & 0\\ 0 & -1\end{matrix}\begin{matrix}I_1\\I_2\end{matrix}=0$
VCVS		$\begin{matrix}0 & 0\\ \mu & -1\end{matrix}\begin{matrix}U_1\\U_2\end{matrix}+\begin{matrix}1 & 0\\ 0 & 0\end{matrix}\begin{matrix}I_1\\I_2\end{matrix}=0$
CCCS		$\begin{matrix}1 & 0\\ 0 & 0\end{matrix}\begin{matrix}U_1\\U_2\end{matrix}+\begin{matrix}0 & 0\\ \beta & -1\end{matrix}\begin{matrix}I_1\\I_2\end{matrix}=0$
CCVS		$\begin{matrix}1 & 0\\ 0 & -1\end{matrix}\begin{matrix}U_1\\U_2\end{matrix}+\begin{matrix}0 & 0\\ r & 0\end{matrix}\begin{matrix}I_1\\I_2\end{matrix}=0$

2.1.2 拓扑矩阵法

用拓扑矩阵法列方程仍然是以 KCL 方程、KVL 方程和 VCR 方程为基础，但最后形成的方程是一个以电路节点电位为变量的节点方程。由于大多数电路的节点数小于支路数，故节点方程组所含独立方程数目比表矩阵法方程组的独立方程数要少得多。

为了便于描述，假定电路中只含有两类元件：(1) 独立电流源。若电路中含有独立电压源，应先将其转换为等效电流源。(2) 用导纳描述的非独立源元件，其支路方程可由 $I_b = Y_b U_b$ 表示。受控源只能是电压控制电流源。这样，我们就可以把所有的支路分为导纳支路和电流源支路两组。支路电流向量 I、支路电压向量 U 以及关联矩阵 A 也均分为相应的两部分，分别用下标 b 和 J 表示，即

$$I = [I_b \quad I_J]^T \qquad U = [U_b \quad U_J]^T \qquad A = [A_b \quad A_J]$$

拓扑矩阵法建立节点方程的步骤如下：

(1) 首先由电路拓扑信息建立关联矩阵 A，以及由 A 表示的 KCL 方程，该 KCL 方程为

$$AI = [A_b \quad A_J] \begin{matrix} I_b \\ I_J \end{matrix} = 0$$

亦即

$$A_b I_b = - A_J I_J \tag{2.1.6}$$

(2) 按照前面所规定的两类元件建立支路特性方程组，即有

$$Y_b U_b = I_b \tag{2.1.7}$$

和

$$I_J = I_S \tag{2.1.8}$$

(3) 将(2.1.7)和(2.1.8)式代入(2.1.6)式，得

$$A_b Y_b U_b = - A_J I_S \tag{2.1.9}$$

(4) 建立由关联矩阵 A 表示的 KVL 方程：

$$U = A^T V_n$$

即

$$\begin{matrix} U_b \\ U_J \end{matrix} = \begin{matrix} A_b^T \\ A_J^T \end{matrix} V_n \tag{2.1.10}$$

这里 U_J 表示独立电流源两端的电压。(2.1.10)式可改写为

$$U_b = A_b^T V_n \tag{2.1.11}$$

$$U_J = A_J^T V_n \tag{2.1.12}$$

(5) 将(2.1.11)式代入(2.1.9)式，得到

$$A_b Y_b A_b^T V_n = - A_J I_S \tag{2.1.13}$$

设 $Y_n = A_b Y_b A_b^T$，$I_{Sn} = - A_J I_S$，则(2.1.13)式变为

$$Y_n V_n = I_{Sn} \tag{2.1.14}$$

这就是节点方程组。其中 Y_n 为节点导纳矩阵，V_n 为节点电位向量，I_{Sn} 为节点电流源向量。(2.1.14)式也称为节点导纳方程组。

例 2.1.2　某电网络如图 2.1.2 所示。用拓扑矩阵法列出该电路的节点导纳方程组。

建立关联矩阵 A:

$$A = [A_b \quad A_J] = \begin{matrix} 1 & 0 & 0 & 1 & 0 & 1 & -1 & 0 & 1 \\ 0 & 1 & 0 & -1 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & -1 & -1 & 0 & -1 & -1 \end{matrix}$$

建立支路导纳矩阵 Y_b:

$$Y_b = \begin{matrix} \frac{1}{R_1} & & & & & & & & \\ & G_2 & & & & & & & \\ & & \frac{1}{SL_3} & & & & & & \\ & & & G_4 & & & & & \\ & & & & SC_5 & & & & \\ & & & & & SC_6 & & & \end{matrix}$$

则节点导纳矩阵为

$$Y_n = A_b Y_b A_b^T = \begin{matrix} \frac{1}{R_1} + G_4 + SC_6 & -G_4 & -SC_6 \\ -G_4 & G_2 + G_4 + SC_5 & -SC_5 \\ -SC_6 & -SC_5 & SC_5 + SC_6 + \frac{1}{SL_3} \end{matrix}$$

建立等效电流源向量 I_{Sn}:

$$I_{Sn} = -A_J I_S = \begin{matrix} 1 & 0 & -1 \\ 0 & 0 & 0 \\ 0 & 1 & 1 \end{matrix} \begin{matrix} I_{S1} \\ I_{S2} \\ I_{S3} \end{matrix} = \begin{matrix} I_{S1} - I_{S3} \\ 0 \\ I_{S2} + I_{S3} \end{matrix}$$

图 2.1.2　例 2.1.2 电路图及拓扑图

综上所述,建立节点方程的关键是形成节点导纳矩阵 Y_n 和等效电流源向量 I_{Sn}。在建立方程的过程中,首先建立关联矩阵 A 和支路导纳矩阵 Y_b,经过一系列的矩阵运算,最终求得 Y_n 和 I_{Sn}。这种由网络拓扑矩阵经过变换求得节点方程的方法是最初网络图论分析中建立网络方程的一种方法。一般 A 和 Y_n 中的非零元很少,用满矩阵处理 A 和 Y_b 显然既费存储容量又费计算时间,不很经济。实际上完全可以直接建立 Y_n 和 I_{Sn},这就是下面要介绍的节点法。

2.1.3 节点法

用节点法列节点方程可以不必建立矩阵 A, Y_b, 而直接形成节点导纳矩阵 Y_n 和等效电流源向量 I_{sn}。为了方便编写程序和用计算机直接写出 Y_n 和 I_{sn}, 我们要研究电路中各种元件对 Y_n 和 I_{sn}的贡献, 掌握各元件的导纳值在 Y_n 中的送值规律, 及独立电流源在 I_{sn}中的送值规律, 便可将它们逐一送入 Y_n 和 I_{sn}的相应位置。为此我们引入“送值表”, 将每个元件对节点导纳矩阵 Y_n 和等效电流源向量 I_{sn}的贡献以表格形式表示出来, 可以一目了然。

1. 无源二端元件对节点方程的贡献

设无源二端元件支路的起节点为 i, 终节点为 j, 元件电导值是 G, 如图 2.1.3 所示。它对节点纳导方程的贡献, 列于表 2.1.3 的送值表中。其中 RHS(right hand side) 表示方程(2.1.14) 的右端向量, 即对应等效电流源向量 I_{sn}。

表 2.1.3 无源二端元件送值表

列 / 行	V_i	V_j	RHS
i	G	- G	
j	- G	G	

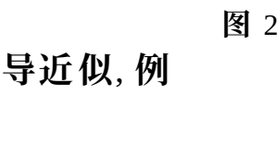


图 2.1.3 无源二端元件支路

对电阻支路, $G = 1/R$。

对电容支路, 直流情况下电容开路, 可用一个小电导近似, 例如 $G = 10^{-7}$S; 在频域分析中, $G = SC = j\omega C$。

对电感支路, 直流情况下电感短路, 可用一个大电导近似, 例如 $G = 10$S; 在频域分析中, $G = 1/SL = -j/\omega L$。

如果二端元件的两个节点中有一个是参考节点, 比如 j 节点是参考节点, 则表 2.1.3 中 j 节点所对应的行列不存在, 该元件在 Y_n 中的贡献只体现在 $Y_n(i, i)$ 上有个 G 值。

无源二端元件对 Y_n 的贡献总是相对矩阵 Y_n 的主对角线对称。

2. 独立源对节点方程的贡献

(1) 独立电流源支路

独立电流源对节点方程组(2.1.14) 的贡献只体现在方程右端 I_{sn}向量中。设电流源的始节点为 i, 终节点为 j, 值为 I_s, 如图 2.1.4 所示, 其送值表如表 2.1.4 所示。

表 2.1.4 独立电流源送值表

列 / 行	V_i	V_j	RHS
i			$-I_s$
j			I_s

图 2.1.4 独立电流源支路

(2) 独立电压源支路

严格地说, 节点法是不能处理理想电压源的, 因为理想电压源

的电导值为无穷大,而且也不能转换为等效电流源。但可以采用下列近似处理方法。

1) 在理想电压源支路中串联一个小电阻(图2.1.5(a))。因为任何一个实际电压源都存在内阻,这样处理符合实际情况,然后将电压源转换为电流源(图(b))来处理。

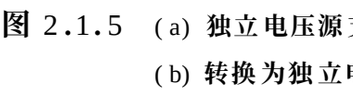


图 2.1.5 (a) 独立电压源支路;
(b) 转换为独立电流源

图 2.1.6 独立电压源转换为独立电流源

2) 在理想电压源支路(图2.1.6(a))中串联电阻+ R 和- R(一般取 R= 1)(图(b)),然后再将电压源转换为电流源(图(c))。在这种情况下,增加了一个节点 i′。

3. 受控源对节点方程的贡献

(1) 电压控制电流源(VCCS)

VCCS 如图 2.1.7。设电压控制支路起、终点为 i、j,受控电流源起、终点为 k、l,控制系数为互导 g_m,量纲是 S(西门子)。VCCS 的支路特性方程是 $I_2= g_mU_1$。由于节点方程中变量是节点电位,相应的电流是节点电流,故支路方程可写为

$$I_2 = g_m(V_i - V_j)$$

$$I_2 = I_k = - I_l$$

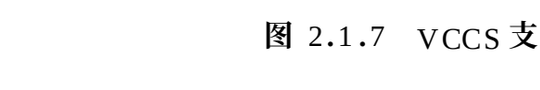


图 2.1.7 VCCS 支路

电流源对节点方程的贡献本应写在节点导纳方程组(2.1.14)右端向量中,由于 $I_2= g_m(V_i - V_j)$ 中,V_i 和 V_j 是未知量,故移到方程组左端系数矩阵 Y_n 中。若在节点 i,j 或 k,l 中有一节点是参考节点,则从送值表 2.1.5 中划去该节点所对应的一行和一列。从送值表上还可看出,VCCS 的送值可能都不在 Y_n 的主对角线上,从而使 Y_n 变为一个非对称矩阵。

表 2.1.5 VCCS 送值表

行 \ 列	V_i	V_j	V_k	V_l	RHS
i					
j					
k	g_m	$-g_m$			
l	$-g_m$	g_m			

(2) 电流控制电流源(CCCS)

CCCS 如图 2.1.8 所示。CCCS 的控制支路是个短路支路,其电导无穷大。故节点法

实际上不能处理 CCCS。但是，在实际电路中如果控制支路不是短路支路，其电导为 G，则可将 CCCS 转化为 VCCS 来处理。这种情况下，设控制支路起、终点为 i、j，受控支路起、终点为 k、l，控制系数是电流放大系数 β，则 CCCS 的特性方程为

$$I_2 = \beta I_1 = \beta G(V_i - V_j)$$

CCCS 的送值表如表 2.1.6 所示。

图 2.1.8　CCCS 支路

表 2.1.6　CCCS 送值表

行＼列	V_i	V_j	V_k	V_l	RHS
i					
j					
k	βG	$-\beta G$			
l	$-\beta G$	βG			

采用节点法的几种元件的送值表列于表 2.1.7 中。

表 2.1.7　采用节点法的元件送值表

元　件	符　号	送　值　表	特性方程
导纳		行＼列 | V_i | V_j | RHS i | G | -G | j | -G | G |	$I_G = G(V_i - V_j)$
电流源		行＼列 | V_i | V_j | RHS i | | | $-I_S$ j | | | I_S	$I_i = -I_j = I_S$
电压控制电流源 VCCS		行＼列 | V_i | V_j i | | j | | k | g_m | $-g_m$ l | $-g_m$ | g_m	$I_2 = g_m(V_i - V_j)$ $I_2 = I_k = -I_l$
电流控制电流源 CCCS		行＼列 | V_i | V_j i | | j | | k | βG | $-\beta G$ l | $-\beta G$ | βG	$I_2 = \beta I_1 = \beta G(V_i - V_j)$ $I_2 = I_k = -I_l$

采用节点法列方程,只要先将电路各节点顺序编号,然后逐个扫视电路中各元件支路,按照各元件送值表的格式,将值填入节点方程的 Y_n 和 I_{Sn} 中,即可形成节点方程组。例如对于前面例题 2.1.2 所示电路及要求,可以很容易用节点法直接列出其节点方程组 $Y_nV_n = I_{Sn}$,即

$$\begin{bmatrix} \frac{1}{R_1} + G_4 + SC_6 & -G_4 & -SC_6 \\ -G_4 & G_2 + G_4 + SC_5 & -SC_5 \\ -SC_6 & -SC_5 & SC_5 + SC_6 + \frac{1}{SL_3} \end{bmatrix} \begin{bmatrix} V_1 \\ V_2 \\ V_3 \end{bmatrix} = \begin{bmatrix} I_{S1} - I_{S3} \\ 0 \\ I_{S2} + I_{S3} \end{bmatrix}$$

节点法的优点是所形成的节点方程组阶数较低,对于含 n 个独立节点的电路,其节点方程组是 n 阶的。列方程的方式比较简单,易于编程。节点方程组的系数矩阵大多具有对角优势。节点法的缺点是不能直接处理独立电压源,零值电阻元件等支路导纳值为无穷大的元器件,也不能直接处理除 VCCS 以外的受控源。另外,支路电流变量不能直接作为节点方程组的未知变量,因而限制了节点法的应用范围。

2.1.4 改进节点法

改进节点法是在节点法的基础上克服了上述节点法所存在的一系列问题而提出的一种改进的节点法。它除了以节点电压作为方程未知变量外,还引入支路电流作为新的未知变量。既保持了节点法方程阶数较低、方法简单的优点,又克服了节点法不能直接处理独立源支路、阻抗为零支路以及流控器件的弱点。因此改进节点法得到了广泛的应用。美国著名的电路模拟程序 SPICE 就是采用改进节点法列方程,目前国内外多数电路 CAD 工具也都采用此法列方程。

我们先用网络拓扑知识介绍改进节点法所列方程组形式,然后像前面所述节点法那样用送值表描述各种元器件对方程组的贡献,从而直接建立改进节点方程。

改进节点法的基本思想,是将元件分为三类:

第一类:用导纳描述的元件。这些元件只需选节点电位作为方程变量,不必选其支路电流作为方程变量。

第二类:不用导纳描述的元件,如独立电压源、VCVS、CCCS、CCVS 等等。此外,还包括那些需要支路电流作为输出变量的元件,如电感、互感元件等等。

第三类:独立电流源。

以上三类元件的特性方程分别为:

$$Y_1U_1 = I_1 \tag{2.1.15}$$

$$Y_2U_2 + Z_2I_2 = E_2 \tag{2.1.16}$$

$$I_3 = I_S \tag{2.1.17}$$

按照这种分组顺序,可将用关联矩阵 A 表示的 KCL 方程、KVL 方程也分为相应的三

部分。

KCL:

$$AI = [A_1 \quad A_2 \quad A_3] \begin{matrix} I_1 \\ I_2 \\ I_S \end{matrix} = 0 \tag{2.1.18}$$

KVL:

$$\begin{matrix} U_1 \\ U_2 \\ U_J \end{matrix} = \begin{matrix} A_1^T \\ A_2^T \\ A_3^T \end{matrix} V_n$$

即

$$U_1 = A_1^T V_n \tag{2.1.19}$$

$$U_2 = A_2^T V_n \tag{2.1.20}$$

$$U_J = A_3^T V_n \tag{2.1.21}$$

将(2.1.18)式改写为

$$A_1 I_1 + A_2 I_2 = - A_3 I_S$$

并将(2.1.15)式代入上式,得

$$A_1 Y_1 U_1 + A_2 I_2 = - A_3 I_S$$

再用(2.1.19)式替换 U_1,得

$$A_1 Y_1 A_1^T V_n + A_2 I_2 = - A_3 I_S \tag{2.1.22}$$

我们设式(2.1.22)中

$$A_1 Y_1 A_1^T = Y_{n1}$$

$$- A_3 I_S = I_{Sn}$$

则(2.1.22)式变为

$$Y_{n1} V_n + A_2 I_2 = I_{Sn} \tag{2.1.23}$$

将(2.1.20)式代入(2.1.16)式,得

$$Y_2 A_2^T V_n + Z_2 I_2 = E_2 \tag{2.1.24}$$

将(2.1.23)和(2.1.24)式合并为一个矩阵方程,即有

$$\begin{matrix} Y_{n1} & A_2 \\ Y_2 A_2^T & Z_2 \end{matrix} \begin{matrix} V_n \\ I_2 \end{matrix} = \begin{matrix} I_{Sn} \\ E_2 \end{matrix} \tag{2.1.25}$$

这就是改进节点方程的一般形式。可以看出,其中 Y_{n1} 是由第一类元件形成的节点导纳矩阵,I_{Sn} 是等效电流源向量,Y_{n1} 和 I_{Sn} 的性质和形成方法,与前面所述用节点法形成节点导纳矩阵 Y_n 和等效电流源向量 I_{Sn} 的方法完全相同。其中 A_2, $Y_2 A_2^T$, Z_2 是电流未知变量关系式矩阵,对多数元件 $Y_2 A_2^T$ 的非零结构与 A_2^T 的相同。V_n 是未知节点电位向量,I_2 是未知支路电流向量,E_2 是已知电压源向量。由于改进节点方程组的未知变量由节点电压和支路电流组成,系数矩阵中的元素既有导纳,又有阻抗及无量纲量,故我们也称该方程组为混合方程组,并简写为

$$TX = B \tag{2.1.26}$$

建立混合方程组 $TX=B$，完全不必像上面所述那样从建立关联矩阵 A 开始，进行一系列矩阵运算而形成。改进节点法与节点法一样，可以直接形成混合方程组。它以元件的送值表为依据，将各元件分别处理，逐一送入元件对系数矩阵 T 及右端向量 B 的贡献，从而形成混合方程组 $TX=B$。在改进节点法所包含的三类元件中，第一类导纳描述元件对方程系数矩阵的贡献，以及第三类电流源元件对右端向量 B 的贡献，与前面介绍的节点法完全相同，改进节点法的关键是如何处理第二类非导纳描述元件。

对于第二类非导纳描述的元件，选取它的支路电流作为新的变量，形成(2.1.25)式中未知向量中的 I_2 向量。根据每个元件的支路特性方程和其支路电流变量，在已经形成的系数矩阵基础上，增加一行一列。新增加的行用来表示元件的特性方程，新增加的列用来表示元件支路电流对节点电流的贡献。某些二端口器件(如 CCVS)需要用两个特性方程描述，则应选取其两个支路电流都作为新添变量，并使原系数矩阵增加两行两列，下面举例说明第二类元件对系数矩阵 T 和右端向量 B 的送值格式。

假定第一、三类元件已填入方程 $TX=B$ 中，形成了 Y_{n1} 和 I_{Sn}。对于有 n 个独立节点的电路，Y_{n1} 为 n 阶方阵，I_{Sn} 为 n 维向量。设已形成的系数矩阵 T 为 m 阶(最初 $m=n$)，则由第二类元件而新增加的一行一列分别为第 $(m+1)$ 行和 $(m+1)$ 列。下面以独立电流源和 VCVS 为例，具体说明第二类元件对混合方程组(2.1.25)的贡献。

(1) 独立电压源

电压源符号如图 2.1.9。设电压源的正负节点分别是 i、j，支路电流为 I_E，则电压源支路方程为

图 2.1.9 独立电压源符号

$$V_i - V_j = E \tag{2.1.27}$$

支路电流与节点电流关系为

$$I_i = I_E \tag{2.1.28}$$

$$I_j = -I_E$$

将支路方程(2.1.27)作为附加方程写在已形成方程组下面新加的一行中，并将支路电流与节点电流关系(2.1.28)作为对已形成系数矩阵的附加列，则形成如表 2.1.8 所示的送值表，表中，BR(branch relation)表示增加的附加方程。

表 2.1.8 电压源送值表

行＼列	V_i	V_j	I_E	RHS
i			1	
j			-1	
BR	1	-1		E

(2) 电压控制电压源(VCVS)

VCVS 符号如图 2.1.10。设 VCVS 的控制支路起终节点为 i、j，受控支路起终节点为 k、l，电压放大倍数为 μ，支路电流变量为 I_2，其支路特性方程为

$$V_k - V_l = \mu(V_i - V_j)$$

或 $$-\mu(V_i - V_j) + (V_k - V_l) = 0$$

以及 $$I_i = I_j = 0$$

$$I_k = -I_l = I_2$$

送值表如表 2.1.9 所示。

运算放大器的简化模型可以用 VCVS 来描述，设其输入正负节点分别为 i、j，输出节点为 l，增益为 K，其支路方程为

$$U_l = K(V_i - V_j)$$

图 2.1.10　VCVS 符号

表 2.1.9　VCVS 送值表

行＼列	V_i	V_j	V_k	V_l	I_2	RHS
i						
j						
k					1	
l					-1	
BR	$-\mu$	μ	1	-1		

运放符号和用 VCVS 表示的模型如图 2.1.11(a)、(b)所示。

图 2.1.11　运算放大器符号和模型

按照上述方法，我们不难得出采用改进节点法的其它各种元件的送值表，在表 2.1.10 中，列出了采用改进节点法处理常用元器件的送值表。

表 2.1.10　采用改进节点法的元件送值表

元件	符　号	送　值　表	特性方程
电流源		行＼列 | V_i | V_j | RHS i | | | $-I_S$ j | | | I_S	$I_i = -I_j = I_S$

续表

元件	符　　号	送　值　表	特性方程
电压源		行\列 | V_i | V_j | I_E | RHS i | | | 1 | j | | | -1 | BR | 1 | -1 | | E	$V_i - V_j = E$ $I_i = -I_j = I_E$
阻抗支路		行\列 | V_i | V_j | I_R | RHS i | | | 1 | j | | | -1 | BR | 1 | -1 | -R |	$V_i - V_j - RI_R = 0$ $I_i = -I_j = I_R$ (R 可以为零值)
受控源 CCCS		行\列 | V_i | V_j | V_k | V_l | I_1 | RHS i | | | | | 1 | j | | | | | -1 | k | | | | | β | l | | | | | $-\beta$ | BR | 1 | -1 | | | |	$V_i - V_j = 0$ $I_i = -I_j = I_1$ $I_k = -I_l = \beta I_1$
受控源 VCCS		行\列 | V_i | V_j | V_k | V_l | RHS i | | | | | | j | | | | | | k | g_m | $-g_m$ | | | | l | $-g_m$ | g_m | | | |	$I_k = -I_l = g_m(V_i - V_j)$ $I_i = -I_j = 0$
受控源 VCVS		行\列 | V_i | V_j | V_k | V_l | I_2 | RHS l | | | | | | | j | | | | | | | k | | | | | 1 | | l | | | | | -1 | | BR | $-\mu$ | μ | 1 | -1 | |	$V_k - V_l = \mu(V_i - V_j)$ $I_k = -I_l = I_2$
受控源 CCVS		行\列 | V_i | V_j | V_k | V_l | I_1 | I_2 | RHS i | | | | | 1 | | | j | | | | | -1 | | | k | | | | | | 1 | | l | | | | | | -1 | | BR_1 | 1 | -1 | | | | | | BR_2 | | | 1 | -1 | -r | | |	$V_i - V_j = 0$ $V_k - V_l = rI_1$ $I_i = -I_j = I_1$ $I_k = -I_l = I_2$

续表

<table>
<tr><th>元件</th><th>符　　号</th><th>送　值　表</th><th>特性方程</th></tr>
<tr><td>变压器</td><td></td><td>
<table>
<tr><th>行 \ 列</th><th>V_i</th><th>V_j</th><th>V_k</th><th>V_l</th><th>I_2</th></tr>
<tr><td>i</td><td></td><td></td><td></td><td></td><td>$-K_2$</td></tr>
<tr><td>j</td><td></td><td></td><td></td><td></td><td>K_2</td></tr>
<tr><td>k</td><td></td><td></td><td></td><td></td><td>1</td></tr>
<tr><td>l</td><td></td><td></td><td></td><td></td><td>-1</td></tr>
<tr><td>BR</td><td>$-K_1$</td><td>K_1</td><td>1</td><td>-1</td><td></td></tr>
</table>
</td><td>$V_k - V_l = K_1(V_i - V_j)$
$I_i = -I_j = -K_2 I_2$
$(K_1 = K_2 = N)$</td></tr>
<tr><td>互感</td><td></td><td>
<table>
<tr><th>行 \ 列</th><th>V_i</th><th>V_j</th><th>V_k</th><th>V_l</th><th>I_1</th><th>I_2</th></tr>
<tr><td>i</td><td></td><td></td><td></td><td></td><td>1</td><td></td></tr>
<tr><td>j</td><td></td><td></td><td></td><td></td><td>-1</td><td></td></tr>
<tr><td>k</td><td></td><td></td><td></td><td></td><td></td><td>1</td></tr>
<tr><td>l</td><td></td><td></td><td></td><td></td><td></td><td>-1</td></tr>
<tr><td>BR_1</td><td>1</td><td>-1</td><td></td><td></td><td>$-SL_1$</td><td>-SM</td></tr>
<tr><td>BR_2</td><td></td><td></td><td>1</td><td>-1</td><td>-SM</td><td>$-SL_2$</td></tr>
</table>
</td><td>$V_i - V_j = SL_1 I_1 + SMI_2$
$V_k - V_l = SMI_1 + SL_2 I_2$
$I_i = -I_j = I_1$
$I_k = -I_l = I_2$</td></tr>
</table>

我们将改进节点法建立方程的基本步骤归纳如下:

1) 先把用导纳描述的第一类元件对系数矩阵 T 的贡献填入到 T 中,形成(2.1.25)式中的导纳子矩阵 Y_{n1}。假定电路的独立节点数为 n,Y_{n1}是 n 阶方阵。

2) 把独立源的贡献送入右端向量 B 中,形成 n 维电流源向量 I_{sn}。

3) 对第二类非导纳描述元件,按照送值表的规律,每处理一个元件,在已形成的系数矩阵基础上至少增加一行一列,形成(2.1.25)式中的 $Y_2 A_2^T V_n + Z_2 I_2 = E_2$。

综合以上步骤,可得改进节点法的混合方程组为

$$\begin{matrix} Y_{n1} & A_2 \\ Y_2 A_2^T & Z_2 \end{matrix} \begin{matrix} V_n \\ I_2 \end{matrix} = \begin{matrix} I_{sn} \\ E_2 \end{matrix}$$

例 2.1.3　某电路及拓扑图如图 2.1.12 所示,用改进节点法列出该电路的混合方程组。

图 2.1.12　例 2.1.3 电路及其拓扑图

根据图中标出的节点号,首先写出电导G_2、G_3和电容C_4、C_5对节点导纳矩阵Y_{n1}的

贡献。然后,根据电压源和 VCVS 的送值表,分别增加一行一列。结果生成的混合方程组为

$$\left[\begin{array}{cccc|cc}
G_2 & -G_2 & 0 & 0 & 1 & 0 \\
-G_2 & G_2+G_3+SC_4 & -G_3 & -SC_4 & 0 & 0 \\
0 & -G_3 & G_3+SC_5 & 0 & 0 & 0 \\
0 & -SC_4 & 0 & SC_4 & 0 & 1 \\
\hline
1 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & -\mu & 1 & 0 & 0
\end{array}\right]
\left[\begin{array}{c} V_{n1} \\ V_{n2} \\ V_{n3} \\ V_{n4} \\ \hline I_E \\ I_7 \end{array}\right]
=
\left[\begin{array}{c} 0 \\ 0 \\ 0 \\ 0 \\ \hline E_1 \\ 0 \end{array}\right] \tag{2.1.29}$$

2.1.5 双图法

双图法,也称为双图改进节点法。顾名思义,它是将由一个拓扑图描述电路拓扑信息的改进节点法改为由两个拓扑图描述的改进节点法,从而消除了改进节点方程组中的一些冗余变量,降低了方程组的阶数。例如,短路支路的电压、开路支路的电流、电流源的电压以及理想电压源的电流等等,如果不是特别需要,我们都可认为这些变量是一些冗余变量。若在建立电路方程的过程中能消除这些变量,无疑会进一步降低方程组的阶数,进而减少了存储量和计算量。

1. V-图与 I-图

双图法列电路方程需采用两个不同的拓扑图。一个图描述支路电压之间的关系,称为 V-图,用来列 KVL 方程;另一个图描述支路电流之间的关系,称之为 I-图,用来列 KCL 方程。建立 V-图和 I-图的一般规则如下:

(1) 如果电路中某支路电流为零,则在 I-图中略去该支路。例如受控源 VCVS 的电压控制支路的电流。

(2) 如果电路中某支路电压为零,则在 V-图中略去该支路。例如受控源 CCCS 和 CCVS 的电流控制支路的电压。

(3) 如果电路中某支路的特性方程中不存在支路电流变量,而且对该支路电流不感兴趣,则可以从 I-图中略去该支路。例如独立电压源支路。

(4) 如果电路中某支路的特性方程中不存在支路电压变量,而且对该支路电压不感兴趣,则可以从 V-图中略去该支路。例如独立电流源支路。

按照上述原则,对同一电路建立的 V-图和 I-图,其结构、节点数和支路数都可能不同,当然其关联矩阵也不相同。

下面我们通过一个例子说明双图法建立 V-图和 I-图的过程。

例 2.1.4　画出图 2.1.13(a)所示电路的 V-图和 I-图。注意:图中○内数字表示原拓扑图节点编号,□内数字表示 I-图的节点编号,△内数字表示 V-图的节点编号。

1) I-图:按照前面所述的规则建立 I-图。

电压源 E_1 的支路方程 $V_{n1}=E_1$ 中没有支路电流变量 I_E,如果我们不需要计算 I_E,那么在 I-图中就没有这条支路 b_1,则原拓扑图中节点①与地节点合并为 I-图中的地节点[0],节点②为 I-图中的节点[1]。

受控源 VCVS 的支路方程是 $U_{b7}=\mu U_{b6}$,控制支路 b_6 的支路电流为零,受控支路 b_7 的

图 2.1.13　例 2.1.4 电路图及其 I-图和 V-图

(a) 电路图; (b) 原拓扑图; (c) I-图; (d) V-图

电流 I_{b7} 没有出现在支路方程中, 故 b_6 和 b_7 都可以从 I-图中去掉。这样 I-图就减少了两个节点, 三条支路, 如图 2.1.13(c)所示。

2) V-图: 按照建立 V-图的规则观察, 电路图中无支路电压为零支路, 也不存在不含电压变量的支路方程。故此电路的 V-图没有可消去的边, 与原拓扑图相同, 如图 2.1.13(d)所示。

利用计算机可以较容易地生成电路的 I-图和 V-图。当然, 与只有一个拓扑图的改进节点法相比较, 双图法在生成两个拓扑图方面编程要复杂一些。

2. 用双图法建立改进节点方程组

以上面例 2.1.4 中电路图和 I-图、V-图为例, 利用 I-图和 V-图列方程的具体步骤如下:

(1) 根据 I-图列出 KCL 方程:

$$A_i I_i = 0$$

即

$$\begin{bmatrix} -1 & 1 & 1 & 0 \\ 0 & -1 & 0 & 1 \end{bmatrix} \begin{bmatrix} I_{b2} \\ I_{b3} \\ I_{b4} \\ I_{b5} \end{bmatrix} = \begin{bmatrix} -I_{b2} + I_{b3} + I_{b4} \\ -I_{b3} + I_{b5} \end{bmatrix} = 0$$

或写成

$$\begin{aligned} -I_{b2} + I_{b3} + I_{b4} &= 0 \\ -I_{b3} + I_{b5} &= 0 \end{aligned} \tag{2.1.30}$$

(2) 根据用导纳描述元件的支路方程, 将(2.1.30)式中的支路电流用其支路电压表示式取代, 得

$$\begin{aligned} -G_2U_{b2}+G_3U_{b3}+SC_4U_{b4}&=0\\ -G_3U_{b3}+SC_5U_{b5}&=0 \end{aligned} \tag{2.1.31}$$

(3) 根据 V-图,用节点电位取代支路电压,得

$$\begin{aligned} -G_2(V_{n1}-V_{n2})+G_3(V_{n2}-V_{n3})+SC_4(V_{n2}-V_{n4})&=0\\ -G_3(V_{n2}-V_{n3})+SC_5V_{n3}&=0 \end{aligned} \tag{2.1.32}$$

(4) 根据 V-图列出非导纳描述元件的特性方程:

$$\begin{aligned} V_{n1}&=E_1\\ V_{n4}&=\mu V_{n3} \end{aligned} \tag{2.1.33}$$

(5) 将(2.1.32)和(2.1.33)式合并成一个方程组,并写成矩阵形式,得

$$\begin{bmatrix} -G_2 & G_2+G_3+SC_4 & -G_3 & -SC_4\\ 0 & -G_3 & G_3+SC_5 & 0\\ 1 & 0 & 0 & 0\\ 0 & 0 & -\mu & 1 \end{bmatrix} \begin{bmatrix} V_{n1}\\ V_{n2}\\ V_{n3}\\ V_{n4} \end{bmatrix} = \begin{bmatrix} 0\\ 0\\ E_1\\ 0 \end{bmatrix} \tag{2.1.34}$$

这就是用双图法列出的节点方程组。

可以看出,采用双图法列电路方程组有如下特点:

1) 由 I-图和 V-图信息所构成的是方程系数矩阵上面的 2×4 维子阵。I-图的独立节点数 2 对应该子阵的行数,V-图的独立节点数 4 对应子阵的列数。

2) 方程组的阶数等于 I-图的独立节点数与 V-图中非导纳描述的支路数之和,阶数为 4。

3) 方程组(2.1.34)与前面例 2.1.3 中用改进节点法所列方程组(2.1.29)相比较,电路相同,但改进节点法所列方程组是 6×6 阶,有非零元 19 个;双图法所列方程组是 4×4 阶,有非零元 9 个。可见双图法通过消减冗余变量,无论在方程阶数还是非零元数目上都少于改进节点法。

3. 计算机自动建立双图改进节点方程组

用双图法自动生成混合方程组的方法与改进节点法类似,主要区别在于双图法是由 I-图和 V-图分别确定矩阵的行号和列号。

设双图法所形成混合方程组与改进节点法形式的基本相同,即具有如下形式:

$$\begin{bmatrix} Y_{n1} & A_2\\ Y_2A_2^T & Z_2 \end{bmatrix} \begin{bmatrix} V_n\\ I_2 \end{bmatrix} = \begin{bmatrix} I_{Sn}\\ E_2 \end{bmatrix}$$

(1) 电路中用导纳描述的元件对系数矩阵贡献,形成了 Y_{n1} 子阵。如果 I-图与 V-的节点数不同,则 Y_{n1} 不是个方阵。I-图独立节点数决定 Y_{n1} 的行数,V-图的独立节点数决定 Y_{n1} 的列数。假定导纳描述元件支路在 I-图中的起终节点号是 i_I、j_I,在 V-图中的起终节点号是 i_V、j_V,则导纳元件对 Y_{n1} 的贡献由表 2.1.11 的送值表体现。若 i_I 或 j_I 为零,则系数矩阵中不存在其对应的行;若 i_V 或 j_V 为零,则系数矩

表 2.1.11 导纳描述元件送值表

行 \ 列	i_V	j_V
i_I	G	-G
j_I	-G	G

阵中不存在其对应的列。

(2) 电路中非导纳描述的元件,按照它们的特性方程,在已经形成的子矩阵(最初为Y_{n1}子阵)的基础上,依次增加一行或一列,从而实现对系数矩阵和右端向量的送值。表2.1.12中列出了一些常用元件的双图法送值表。

表 2.1.12　采用双图法的元件送值表

元件	符号	送　值　表	特性方程
电压源		m+1 | 行\列 | i_V | j_V | RHS BR | 1 | -1 | E	$V_{iV}-V_{jV}=E$
短路支路		行\列 | I_E i_I | 1 j_I | -1	I_E 为任意值
VCCS		行\列 | i_V | j_V k_I | g_m | $-g_m$ l_I | $-g_m$ | g_m	$I_{kI}=g_m(V_{iV}-V_{jV})$ $I_{lI}=-g_m(V_{iV}-V_{jV})$
VCVS		行\列 | i_V | j_V | k_V | l_V BR | $-\mu$ | μ | 1 | -1	$(V_{kV}-V_{lV})-\mu(V_{iV}-V_{jV})=0$
CCCS		行\列 | I_1 i_I | 1 j_I | -1 k_I | β l_I | $-\beta$	$I_{iI}=-I_{jI}=I_1$ $I_{kI}=-I_{lV}=\beta I_1$
CCVS		行\列 | k_V | l_V | I_1 i_I | | | 1 j_I | | | -1 BR | 1 | -1 | -r	$V_{kV}-V_{lV}=rI_1$

本章介绍了五种常用的建立电路方程组的方法。这些方法都是以网络拓扑原理和元件支路特性方程为基础,用计算机自动建立矩阵形式的电路方程组。由于不同方法选用的变量性质和数目不同,因而形成方程组的阶数和形式也大不相同。我们以例2.1.3电路为例,对三种方法所列方程的阶数及矩阵密度进行比较,如表2.1.13所示。

可以看出,表矩阵法所列方程阶数最高,矩阵密度最低,需用稀疏矩阵技术求解,即使对于规模很小的电路也是如此。但表矩阵法形成矩阵方便,对电路描述全面,并且经过变换也能得到较为紧凑的形式,故有些实用程序也采用表矩阵法。改进节点法形成的方程组阶数较低,矩阵所含非零元数不多。随着电路规模的加大,矩阵的稀疏性还会进一步加大,

表 2.1.13　列方程方法比较

列方程方法	矩阵维数	非零元个数	矩阵密度
表矩阵法	18× 18	39	12%
改进节点法	6× 6	15	41.7%
双图改进节点法	4× 4	9	56%

如果采用稀疏矩阵技术进行处理,存储量和计算量都会大大减少,因此是目前应用最广泛的一种方法。双图改进节点法进一步降低了方程组的阶数,非零元数也显著地减少,但由于列方程时要附加拓扑图的运算,编程略显复杂一些。双图法在处理含有较多受控源或短路、开路支路的电路,其效果较为明显,尤其是处理开关电容电路。

改进节点法是目前应用最为广泛的列方程方法,在 SPICE 程序中就是采用改进节点法列电路方程。在以后各章节中我们都采用改进节点法来建立电路方程组。

2.2　线性代数方程组的数值解法

2.1 节介绍了如何用计算机自动建立电路方程组,这一节讲述如何求解方程组。无论是用表矩阵法列出的表矩阵方程组,或是用节点法列出的节点导纳方程组,还是由改进节点法列出的混合方程组,当用于不同的分析领域时,电路方程组的形式是不同的。进行线性稳态分析时,方程组是线性代数方程;进行频域分析时,方程组是复数代数方程组;进行非线性直流分析时,方程组是非线性代数方程组;进行非线性时域分析时,是非线性常微分方程组。但是,不论是什么形式的电路方程组,最后都要转换为实系数线性代数方程来求解。因此,实系数线性代数方程组的求解问题在电路 CAD 中占有相当重要的地位,求解效率的高低、存储量的大小、计算速度的快慢、计算精度的高低等等都直接影响着电路分析的质量。

电路 CAD 中常用的解方程方法是高斯消去法和 LU 分解法,我们简单地介绍这两种算法的计算公式和计算步骤,对于稀疏矩阵技术在电路方程组数值解法中的重要意义和基本算法也做一些简要的说明。

2.2.1　高斯消去法

1. 高斯消去法的基本步骤

线性代数方程组的一般形式为

$$\begin{bmatrix} t_{11} & t_{12} & \cdots & t_{1n} \\ t_{21} & t_{22} & \cdots & t_{2n} \\ \vdots & \vdots & \vdots & \vdots \\ t_{n1} & t_{n2} & \cdots & t_{nn} \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{bmatrix} = \begin{bmatrix} b_1 \\ b_2 \\ \vdots \\ b_n \end{bmatrix} \tag{2.2.1}$$

或

$$TX = B \tag{2.2.2}$$

高斯消去法包括正消和回代两大步骤。通过正消把系数矩阵 T 化为上三角矩阵,然后通过回代逐一求得未知数 x_n, x_{n-1}, …, x_2, x_1。

(1) 正消步骤

对(2.2.1)式的增广矩阵,其正消过程如下。

1) 第 1 次正消

设 $t_{11}^{0}\neq 0$,将增广矩阵第一行各元素除以主元 t_{11}^{0},即

$$t_{1j}^{1}=t_{1j}^{0}/t_{11}^{0}\qquad b_{1}^{1}=b_{1}^{0}/t_{11}^{0}\qquad (j=1,2,\cdots,n)$$

其中上标“0”表示矩阵元素的原始数据,“1”表示该元素经过第一次正消处理后的数值。

为了使第一列中主元以下的各元素变为零,我们对第 2 至 n 行施行下列变换:

$$t_{ij}^{1}=t_{ij}^{0}-t_{i1}^{0}t_{1j}^{1}\qquad (i=2,3,\cdots,n;\ j=2,3,\cdots,n)$$

经过第一次正消处理后,增广矩阵变化如下:

$$\begin{bmatrix} t_{11}^{0} & t_{12}^{0} & \cdots & t_{1n}^{0} & b_{1}^{0}\\ t_{21}^{0} & t_{22}^{0} & \cdots & t_{2n}^{0} & b_{2}^{0}\\ \vdots & \vdots & \vdots & \vdots & \vdots\\ t_{n1}^{0} & t_{n2}^{0} & \cdots & t_{nn}^{0} & b_{n}^{0} \end{bmatrix}\xrightarrow{\text{第一次正消}}\begin{bmatrix} 1 & t_{12}^{1} & t_{13}^{1} & \cdots & t_{1n}^{1} & b_{1}^{1}\\ 0 & t_{22}^{1} & t_{23}^{1} & \cdots & t_{2n}^{1} & b_{2}^{1}\\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots\\ 0 & t_{n2}^{1} & t_{n3}^{1} & \cdots & t_{nn}^{1} & b_{n}^{1} \end{bmatrix}$$

2) 第 k 次正消($1\leq k\leq n$)

设 $t_{kk}^{k-1}\neq 0$,正消过程的一般表达式为

$$\begin{aligned} t_{kj}^{k}&=t_{kj}^{k-1}/t_{kk}^{k-1}\\ t_{ij}^{k}&=t_{ij}^{k-1}-t_{ik}^{k-1}t_{kj}^{k}\\ b_{i}^{k}&=b_{i}^{k-1}-t_{ik}^{k-1}b_{k}^{k} \end{aligned}\qquad (2.2.3)$$

其中:$k=1,2,\cdots,n$;$i=k+1,k+2,\cdots,n$;$j=k+1,k+2,\cdots,n$。

经过第 k 次正消处理后,增广矩阵变为

$$\left[\begin{array}{ccc|ccc|cc} 1 & t_{12}^{1} & t_{13}^{1} & \cdots & \cdots & \cdots & t_{1n}^{1} & b_{1}^{1}\\ 0 & 1 & t_{23}^{2} & \cdots & \cdots & \cdots & t_{2n}^{2} & b_{2}^{2}\\ \vdots & 0 & \text{W} & \cdots & \cdots & \cdots & \vdots & \vdots\\ \hline \vdots & \vdots & 0 & 1 & t_{k,k+1}^{k} & \cdots & t_{kn}^{k} & b_{k}^{k}\\ \vdots & \vdots & \vdots & 0 & t_{k+1,k+1}^{k} & \vdots & \vdots & \vdots\\ \vdots & \vdots & \vdots & \vdots & \vdots & \text{W} & \vdots & \vdots\\ 0 & 0 & 0 & 0 & t_{n,k+1}^{k} & \cdots & t_{nn}^{k} & b_{n}^{k} \end{array}\right]$$

3) 第 n 次正消

经过 n 次正消($k=n$)后,使最后一个主元 $t_{nn}^{n}=1$,则系数矩阵 T 变为一个主对角线元素都是单位 1 的上三角矩阵,即增广矩阵为

$$\begin{bmatrix} 1 & t_{12}^{1} & t_{13}^{1} & t_{14}^{1} & \cdots & t_{1n}^{1} & b_{1}^{1}\\ 0 & 1 & t_{23}^{2} & t_{24}^{2} & \cdots & t_{2n}^{2} & b_{2}^{2}\\ 0 & 0 & 1 & t_{34}^{3} & \cdots & t_{3n}^{3} & b_{3}^{3}\\ 0 & 0 & 0 & 1 & \cdots & t_{4n}^{4} & b_{4}^{4}\\ \vdots & \vdots & \vdots & \vdots & \text{W} & \vdots & \vdots\\ 0 & 0 & 0 & 0 & \cdots & 1 & b_{n}^{n} \end{bmatrix}\qquad (2.2.4)$$

此时,正消过程完毕。

(2) 回代步骤

经过 n 次正消得到上三角阵后,式(2.2.1)变为如下形式:

$$\begin{bmatrix} 1 & t_{12}^{1} & t_{13}^{1} & \cdots & \cdots & t_{1n}^{1} \\ & 1 & t_{23}^{2} & \cdots & \cdots & t_{2n}^{2} \\ & & 1 & & & t_{3n}^{3} \\ & & & \ddots & & \vdots \\ 0 & & & & \ddots & t_{n-1,n}^{n-1} \\ & & & & & 1 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ \vdots \\ x_{n-1} \\ x_n \end{bmatrix} = \begin{bmatrix} b_1^1 \\ b_2^2 \\ b_3^3 \\ \vdots \\ b_{n-1}^{n-1} \\ b_n^n \end{bmatrix} \tag{2.2.5}$$

由此式的最后一个方程,可直接求出 $x_n = b_n^n$;代入第(n-1)个方程,可求出 x_{n-1};逐一按照此逆序可求解出 x_{n-2}, x_{n-3}, …, x_2, x_1。

回代步骤的一般表达式为

$$\begin{aligned} x_k &= b_k^k - \sum_{j=k+1}^{n} t_{kj}^k x_j \qquad (k = n-1, \cdots, 1) \\ x_n &= b_n^n \end{aligned} \tag{2.2.6}$$

高斯消去法所需的乘除运算次数为$(n^3/3) + n^2 - (n/3)$。

2. 高斯主元素消去法

在高斯消去法的正消过程中,第 k 步正消需要将主对角元素 t_{kk} 去除矩阵同行各元素,这时可能出现两种情况:

(1) 如果主元 t_{kk} 为零,则正消过程无法进行。

(2) 如果主元 t_{kk} 的绝对值很小,由于计算机的机器字长有限,用 t_{kk} 作除数会导致其它元素值数量级的严重增长和舍入误差的扩散,使计算解不可靠。

我们以一个简单的二阶方程组为例,说明主元数值的大小对计算结果可靠性的影响。例如方程组

$$\begin{bmatrix} -0.001 & 6 \\ 2.5 & 5 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} = \begin{bmatrix} 6.001 \\ 2.5 \end{bmatrix} \tag{2.2.7}$$

的精确解为 $x_1 = -1$, $x_2 = 1$。我们以 5 位有效数字进行消元运算,其过程如下:

取主元 $t_{11} = -0.001$,则有

$$\begin{bmatrix} 1 & -6000 \\ 0 & 15005 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} = \begin{bmatrix} -6001 \\ 15004 \end{bmatrix}$$

计算结果为 $x_1 = -1.5$, $x_2 = 0.99993$,与精确解相差甚远。它是因为计算时先用了小主元 -0.001,使约化后的方程元素值的数量级大大增长,经舍入计算,使消元后的三角方程组失去了准确性。如果我们把方程对换一下,改变消元顺序,即有

$$\begin{bmatrix} 2.5 & 5 \\ -0.001 & 6 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} = \begin{bmatrix} 2.5 \\ 6.001 \end{bmatrix} \tag{2.2.8}$$

选 $t_{11} = 2.5$ 作主元,则

$$\begin{bmatrix} 1 & 2 \\ 0 & 6.002 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} = \begin{bmatrix} 1 \\ 6.002 \end{bmatrix}$$

计算结果为 $x_1 = -1$, $x_2 = 1$,这组解即为精确解。这个例子说明,在采用高斯消去法解方

程时,小主元可能产生麻烦,造成解的误差,故应避免采用绝对值小的主元素 t_{kk}^{k}。

在电路 CAD 中,电路各元件的电导值差异很大,因而电路方程系数矩阵 T 中各元素的数值相差悬殊,可能达到 10^8 量级以上。因此,必须采取有效的方法解决这个问题。

目前解决此问题的途径有两个:

1) 选择绝对值较大的系数矩阵元素作主元,进行消元运算,我们称之为选主元消去法。

2) 采用双精度字长运算,其字长是单精度的 2 倍。由于字长增加了一倍,误差的影响将大大减小,可以获得足够精确的解。

下面介绍几种常用的主元消去法。

(1) 列主元消去法

这是一种按矩阵列选取主元的消去法。它是在第 k 步正消前,先比较 k 列中,k 行以后各元素 t_{kk}^{k-1}, $t_{k+1,k}^{k-1}$, …, $t_{n,k}^{k-1}$,从中选出绝对值最大者,作为第 k 步的主元。设主元在第 l 个方程,即

$$\left|t_{lk}^{k-1}\right| = \max_{k\le i\le n}\left|t_{ik}^{k-1}\right|$$

将第 l 个方程与第 k 个方程互换位置,然后再进行消元。这种方法称之为列主元消去法。

(2) 行主元消去法

这是一种按矩阵行选取主元的消去法。它与列主元消去法类似,在第 k 步正消前,先比较第 k 行中,k 列以后各元素 t_{kk}^{k-1}, $t_{k,k+1}^{k-1}$, …, $t_{k,n}^{k-1}$,从中选出绝对值最大者为第 k 步主元。设主元在第 l 列,即

$$\left|t_{kl}^{k-1}\right| = \max_{k\le j\le n}\left|t_{kj}^{k-1}\right|$$

将第 l 列元素与第 k 列元素互换,同时必须将未知量 x_l 与 x_k 也互换。这种方法称之为行主元消去法,它没有列主元消去法应用得普遍,这两种方法都称为部分主元消去法。

(3) 全主元消去法

这是一种在整个系数矩阵中选取主元的消去法。在第 k 步正消前,先比较第 k 行、k 列右下部子矩阵的所有元素,从中选出最大者做为第 k 步主元。即

$$\left|t_{kk}^{k-1}\right| = \max_{\substack{k\le i\le n\\ k\le j\le n}}\left|t_{ij}^{k-1}\right|$$

若该元素位于第 l 行、m 列,则方程需进行相应的行、列交换。这种方法称为全主元消去法。就数值计算的稳定性看,全主元法比部分主元法优越,但计算工作量稍大。

2.2.2 LU 分解法

LU 分解法是高斯消去法的一种变型算法。它将方程组 TX= B 的系数矩阵 T 分解为两个三角矩阵的乘积,即

$$T = LU = \begin{bmatrix} l_{11} & & & \\ l_{21} & l_{22} & & 0 \\ \vdots & \vdots & & W \\ l_{n1} & l_{n2} & \cdots & l_{nn} \end{bmatrix} \begin{bmatrix} 1 & u_{12} & u_{13} & \cdots & u_{1n} \\ & 1 & u_{23} & \cdots & u_{2n} \\ & & 1 & \cdots & \vdots \\ 0 & & & W & u_{n-1,n} \\ & & & & 1 \end{bmatrix} \tag{2.2.9}$$

其中 L 为下三角矩阵,U 为主对角元素为单位 1 的上三角矩阵。而且引入了中间变量 y。于是 TX= B 式可改写为

$$LUX = B \tag{2.2.10}$$

再改写为

$$Ly = B \tag{2.2.11}$$

$$UX = y \tag{2.2.12}$$

先由(2.2.11)式求出 y,再代入(2.2.12)式即可求解出方程组的解 X。式(2.2.11)和式(2.2.12)的求解步骤也称为向前-向后代换。

LU 分解法的一般公式如下。

L 矩阵各元素

$$l_{kj} = t_{kj} - \sum_{p=1}^{j-1} l_{kp} u_{pj} \qquad \begin{matrix} j = 1, 2, \cdots, n \\ k = j, j+1, \cdots, n \end{matrix} \tag{2.2.13}$$

U 矩阵中各元素(当 $l_{jj} \neq 0$ 时)

$$u_{jk} = \left(t_{jk} - \sum_{p=1}^{j-1} l_{jp} u_{pk}\right) \Big/ l_{jj} \qquad \begin{matrix} j = 1, 2, \cdots, n \\ k = j+1, j+2, \cdots, n \end{matrix} \tag{2.2.14}$$

y 向量中各分量(当 $l_{jj} \neq 0$ 时)

$$y_j = \left(b_j - \sum_{p=1}^{j-1} l_{jp} y_p\right) \Big/ l_{jj} \qquad (j = 1, 2, \cdots, n) \tag{2.2.15}$$

X 向量中各分量

$$X_j = y_j - \sum_{p=j+1}^{n} u_{jp} X_p \qquad (j = n, n-1, \cdots, 1) \tag{2.2.16}$$

LU 分解法的总乘除运算量为$(n^3/3) + n^2 - (n/3)$,与高斯消法的相同。在选主元方面,LU 分解法也有同样要求,并有类似的部分主元和全主元方法。

LU 分解法的优点是,当需要反复多次求解方程组 TX= B 时,如果系数矩阵 T 的数值不变,而仅仅是 B 向量的数值变化,则第一次由 T 分解求得的 L、U 阵可以重复使用,省去了 LU 分解所需的运算量$(n^3/3) - (n/3)$,仅需计算 Ly= B 和 UX= y,其运算量只有 n^2 次,因而大大地减少了运算量。在电路 CAD 中,频域分析、时域分析、灵敏度分析以及优化设计等等都需要多次求解电路方程,而且在不少情况下 T 矩阵的数值是不变的。例如用伴随网络法求灵敏度时,伴随网络系数矩阵是原网络系数矩阵 T 的转置矩阵,利用 LU 分解法可以省去伴随网络系数矩阵的 LU 分解运算,是十分有利的。故在电路 CAD 领域中,LU 分解法的应用更为普遍。

应当指出,以上估计高斯消去法和 LU 分解法的乘除运算量时,都是按系数矩阵 T 是满矩阵计算的。实际上对于稍大一些的电路(n> 40),T 矩阵是个稀疏矩阵,矩阵中含有大量的零元素。因此大量的乘除运算是在零元素中进行的,很不合算。如果零元素不存储,也不参加运算,那么存储量和乘除运算量就不再是与 n^3、n^2 成比例,而仅仅是与 n 本身成比例,显然可以节省很多内存和机时。这就是我们下面要介绍的稀疏矩阵技术。

2.2.3 稀疏矩阵技术

1. 稀疏矩阵技术的必要性

在电路 CAD 中,求解线性代数方程组 TX= B 是一个基本问题。无论是线性还是非线性电路,无论是频域还是时域分析,最终都归结为求解 TX= B。在频域分析中,每个频率点要计算一次线性代数方程组;在非线性分析中,每迭代一次要求解一次线性代数方程组;在非线性时域分析中,每个时间点上都要进行多次迭代,在整个计算的时间内则需成百上千次地求解方程;蒙特卡罗统计分析和优化设计所需求解方程组的次数就更多了。因此,如何有效地解线性代数方程组是电路 CAD 的一个重要问题,直接影响着电路 CAD 的效率和质量。

所谓稀疏矩阵,是指含有大量零元素,非零元素在全部矩阵元素中所占百分比很小的矩阵。人们考虑可以利用稀疏矩阵内含有大量零元素的特点,采取措施只存储非零元素和只对非零元素进行运算,以减小存储量和运算量。这类方法称之为稀疏矩阵技术。电路方程组的系数矩阵 T,除了极为简单的情况外,大多是个稀疏矩阵。采用稀疏矩阵技术处理电路方程是非常必要的。电路方程组一般具有如下特点:

(1) 电路方程的系数矩阵通常是个稀疏矩阵。而且随着电路规模的增加,矩阵的阶数增高,稀疏程度也越大。例如 10 个节点的电路,系数矩阵的非零元素可能占 50% 左右;而 100 个节点的典型电路中,非零元素仅占 5% 左右。据统计,对多数实际电路,其矩阵中的非零元数目在 4n 到 6n 之间(n 为方程阶数)。

(2) 电路方程的系数矩阵具有较好的对称性。当电路中不含受控源时,系数矩阵是个对称阵。一般电路中受控源大多以晶体管的等效模型形式出现,这时系数矩阵虽不再保持数值上的对称,但在结构上可能仍然对称,或者仅在矩阵中增补几个非零元便可保证其结构的对称性。

(3) 对一给定电路,当节点编号确定后,则系数矩阵中非零元分布的结构也相应确定,在以后的数值求解过程中,矩阵中元素值可以一次次地变化,但矩阵的非零元结构在求解过程中始终不变。

(4) 电路方程组的系数矩阵中含有大量的+ 1 和- 1 元素。对+ 1 元素的操作也属于空操作,而对- 1 元素进行操作只须改变符号。而矩阵中+ 1 和- 1 元素的分布位置,在多次求解过程中也始终不变。

因此,如果能充分利用电路方程的这些特点,采用稀疏矩阵技术编制出高效率的线性代数方程组的求解程序是非常有意义的。稀疏矩阵技术研究和解决的主要问题是:

(1) 减少存储量

1) 通常只存储非零元素。如果矩阵是对称矩阵,存储量还能进一步减小。

2) 采用优化排序方法,减少消元过程中增加的新的非零元的填入,防止破坏矩阵的稀疏性和增加存储量。

(2) 提高计算速度

1) 只进行非零元素的运算,避免无效的零运算。

2) 采用合理的数据存储技术,使运算过程中对数据信息的分类、检索、插入和删除等

处理尽可能方便,以提高数据检索效率,减少运算时间和存储量。

3) 用行、列调换和优化主元排列方法减少乘除运算次数。

(3) 提高精度,保证数值稳定性

稀疏矩阵技术一般选定了一种较优的主元的消元过程后,就不再改变。实际上是一种不选主元的消元过程。常采用双精度运算,适时选主元或主元容限的方法,提高解的精度并保证数值稳定性。

当然这三个问题的解决既是互相依存的,又可能互相矛盾,往往需要综合权衡,在存储容量、计算时间和数值稳定性之间采取适当的折衷方案。

2. 稀疏矩阵的存储技术

充分利用矩阵的稀疏性,只存储非零元素,可以节省大量内存。但是稀疏矩阵的存储技术所要解决的不仅仅是减少存储容量问题,还要考虑存取时间,检索、查找方便等因素。因为在稀疏矩阵求解过程中,矩阵中非零元结构会发生变动,可能增加或删除某些非零元,因此就有检索、存取、插入或删去非零元是否方便的问题,即存取效率问题。稀疏矩阵的存储方法有许多种,不同的方法所需的存储容量不同,存取效率也不相同。实践中往往更看重它的存取效率,而且会为提高存取效率而牺牲一些存储容量。在这里我们将介绍几种常用的存储方法,并比较它们在存储容量和存取效率等方面的优劣。

(1) 线性表存储方法

线性表存储方法是指在整个消去过程中,矩阵中非零元素按列或按行存储,同一行或同一列的非零元顺序存放在连续的单元中。这里我们介绍其中一种按行存储方法。

最简单的按行线性表存储方法由三个数组构成:RE,ICOL 和 IP,它们的功能如下:

RE(j)存放的是系数矩阵中按行排列的第 j 个非零元的数值;

ICOL(j)存放的是系数矩阵中按行排列的第 j 个非零元的列号;

IP(i)是行首指示器,通常占 n 个单元(n 为矩阵阶数),存放系数矩阵中第 i 行第一个非零元素在数组 RE 中的位置序号。

下面以一个四维的系数矩阵 T 为例,说明算法的应用。

例 2.2.1 设

$$T=\begin{bmatrix} t_{11} & 0 & t_{13} & t_{14} \\ t_{21} & t_{22} & 0 & 0 \\ 0 & 0 & t_{33} & 0 \\ 0 & t_{42} & 0 & t_{44} \end{bmatrix} \tag{2.2.17}$$

RE,ICOL,IP 数组的存储情况如表 2.2.1、表 2.2.2 所示。由这两个表查找矩阵 T 中某一行的非零元素十分方便。例如,要找第二行的元素,由 IP(3) - IP(2) = 2,得知第二行有 2

表 2.2.1 RE,ICOL 数组

序号 / 数组	1	2	3	4	5	6	7	8
RE	t_{11}	t_{13}	t_{14}	t_{21}	t_{22}	t_{33}	t_{42}	t_{44}
ICOL	1	3	4	1	2	3	2	4

个非零元素。因为 IP(2) = 4, 故这两个非零元存放在 RE(4) 和 RE(4+ 1) 中, 其相应的列号存放在 ICOL(4) 和 ICOL(5) 中。

表 2.2.2　行首指示器

数组 \ 序号	1	2	3	4
IP	1	4	6	7

如果 T 为 n 阶方阵, 含有 τ个非零元素 ($\tau < n^2$), 则线性表存储法所需存储量为 2τ + n。这种方法的优点是存储单元比较节约, 数据检索也十分方便。缺点是处理插入新的非零元比较困难。比如在 T 中 t_{23}的位置有一新插入元素, 则 t_{23}应插在 t_{33}前面, 即占据 RE(6) 的位置; 那么 RE(6) 和 ICOL(6) 以后的各元素都要顺序向后移一个单元, 相应的 IP 指针也要变动。如果新插入的非零元数目不多, 采用线性表存储方法还是有优越性的。

若系统矩阵 T 的结构是对称的, 上述方法还可以减化, 存储量也可能进一步降低。结构对称矩阵的特点是, 若有 $t_{ij} \neq 0$, 则一定有 $t_{ji} \neq 0$。根据此特点我们设置三个一维数组存放矩阵非零元的数值, 一个数组存放非零元行(或列)号, 再有一个数组作行首指示器。这些数组和它们的功能如下:

LRE(j) 存放矩阵下三角中按列排列的第 j 个非零元素的数值;

DIAG(j) 存放主对角元素;

URE(j) 存放矩阵上三角中按行排列的第 j 个非零元素的数值;

ICORO(j) 存放元素 URE(j) 的列号, 也即元素 LRE(j) 的行号;

IP(i) 存放上三角元中第 i 行第一个非零元在 URE 中的位置, 该位置同时也是下三角元中第 i 列第一个非零元在 LRE 中的位置。

例 2.2.2　列出对称矩阵 T 的线性表存储方案。设对称矩阵 T 为

$$T = \begin{pmatrix} t_{11} & 0 & t_{13} & 0 \\ 0 & t_{22} & 0 & t_{24} \\ t_{31} & 0 & t_{33} & t_{34} \\ 0 & t_{42} & t_{43} & t_{44} \end{pmatrix}$$

表 2.2.3～表 2.2.5 中列出了上述几个数组的存储情况。

表 2.2.3　URE, LRE 和 ICORO 数组

数　组 \ 序　号	1	2	3
URE	t_{13}	t_{24}	t_{34}
LRE	t_{31}	t_{42}	t_{43}
ICORO	3	4	4

表 2.2.4　DIAG 数组

数组 \ 序号	1	2	3	4
DIAG	t_{11}	t_{22}	t_{33}	t_{44}

表 2.2.5　行列首指示器

数组 \ 序号	1	2	3
IP	1	2	3

可以看出,结构对称矩阵的线性表存储方法节省了存放行列信息(ICORO)所需的存储单元。如果矩阵 T 在数值上也是对称的,那么 LRE 数组可以取消,节省的存储单元就更多了。

(2) 单链存储方法

单链存储法指在消去过程中,用一条链将同一行或同一列的元素信息连接在一起,分别称为按行或按列的单链存储方法。以按列存储为例,这种方法要求设置的四个数组和功能如下:

CE(j)按列存放第 j 个非零元数值;

IROW(j)按列存放第 j 个非零元的行号;

NEXT(j)存放同一列中下一个非零元素的位置;

IP(i)是列首指示器。

表 2.2.6 和表 2.2.7 列出了式(2.2.17)中矩阵 T 的单链存储形式中各数组的存储情况。

表 2.2.6 CE,IROW 和 NEXT 数组

序号 / 数组	1	2	3	4	5	6	7	8	9*
CE	t_{11}	t_{21}	t_{22}	t_{42}	t_{13}	t_{33}	t_{14}	t_{44}	t_{23}
IROW	1	2	2	4	1	3	1	4	2
NEXT	2	0	4	0	6	0	8	0	0

表 2.2.7 列首指示器

序号 / 数组	1	2	3	4
IP	1	3	5	7

单链存储法检索非零元非常方便。例如,由 IP(3)= 5,给出矩阵 T 的第 3 列的第 1 个非零元素在表 2.2.6 中的位置为 5,则它的数值是 CE(5)= t_{13},它的行号为 IROW(5)= 1,这列中下一个非零元的位置放在 NEXT(5)中;由 NEXT(5)= 6,可知这一列第 2 个非零元在该表中位于位置 6。如此可逐个查找所需非零元信息。若 NEXT(j)= 0,说明CE(j)是这一列中最后一个非零元数值。单链法与线性表存储法相比,多用了一个 NEXT 数组,因此对于含有 τ个非零元素的 n 阶方阵 T,采用单链法存储所需的存储单元是(3τ+ n)个,比线性表存储法所需存储量大些。但它的最大优点是插入新的非零元素很容易。假定新插入的非零元是 t_{23},它是第 3 列元素,由 IP(3)= 5 得知第 3 列第 1 个非零元素在表 2.2.6 中位置为 5,并由 NEXT 数组查到第 3 列最后一个元素位于该表的位置 6。因此为插入 t_{23},表 2.2.6 应作如下局部修改:

NEXT(6)= 9

CE(9)= t_{23}

IROW(9)= 2

NEXT(9)= 0

即将原第 3 列最后一个非零元 t_{33}的 NEXT 指针指向新插入的 t_{23}所在的最后一个单元位置 9;NEXT(9) = 0 说明表 2.2.6 中位置 9 存的是第 3 列最后一个非零元(按出现先后次序排列)。可以看出,单链存储法只需修改和增加几个单元,就可实现非零元的插入或删除,比线性表存储法优越。

(3) 双链存储方法

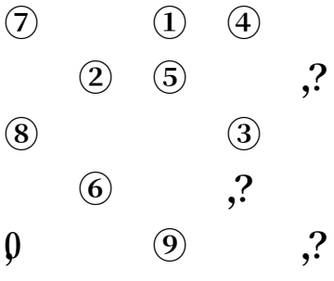
稀疏矩阵技术中的一些计算方法在确定主元序列时,既需要按行、也需要按列检索、删除、插入非零元素。双链存储方法正是适应这一需求的存储方法,这种方法虽然花费存储量较多,但计算效率较高。

双链存储的特点是它不但记录各元素所在的行、列,而且记录各元素上、下、左、右各元素号。对每个非零元素赋以 7 个特征,这 7 个特征是: 1) 行号; 2) 列号; 3) 上邻元素; 4) 下邻元素; 5) 左邻元素; 6) 右邻元素; 7) 元素数值。其中第 3)、4)、5)、6) 特征是指其相邻元素的序号,若无则记为零。

例 2.2.3 列出矩阵 T 的双链表存储方案,设

$$T=\begin{pmatrix} t_{11} & 0 & t_{13} & t_{14} & 0 \\ 0 & t_{22} & t_{23} & 0 & t_{25} \\ t_{31} & 0 & 0 & t_{34} & 0 \\ 0 & t_{42} & 0 & t_{44} & 0 \\ t_{51} & 0 & t_{53} & 0 & t_{55} \end{pmatrix}$$

给矩阵 T 中各非零元素编号,次序任意,则有



将此矩阵列成的双链表格如表 2.2.8 所示。

表 2.2.8 双链表

元素序号 i	行 ROW(i)	列 COL(i)	上邻 UP(i)	下邻 DOWN(i)	左邻 LEFT(i)	右邻 RIGHT(i)	数值
1	1	3	0	5	7	4	t_{13}
2	2	2	0	6	0	5	t_{22}
3	3	4	4	12	8	0	t_{34}
4	1	4	0	3	1	0	t_{14}
5	2	3	1	9	2	11	t_{23}
6	4	2	2	0	0	12	t_{42}
7	1	1	0	8	0	1	t_{11}
8	3	1	7	10	0	3	t_{31}
9	5	3	5	0	10	13	t_{53}
10	5	1	8	0	0	9	t_{51}

续表

元素序号 i	行 ROW(i)	列 COL(i)	上邻 UP(i)	下邻 DOWN(i)	左邻 LEFT(i)	右邻 RIGHT(i)	数值
11	2	5	0	13	5	0	t_{25}
12	4	4	3	0	6	0	t_{44}
13	5	5	11	0	9	0	t_{55}

除了表 2.2.8 中的 7 个数组外,还需设置二个长为 n(n 为方程阶数)的指针数组,分别记录各行和各列的第一个非零元素的序号,如表 2.2.9 所示。

表 2.2.9　行列首指示器

行列号	1	2	3	4	5
行首元素	7	2	8	6	10
列首元素	7	2	1	4	11

显然双链存储方法对处理非零元素的检索、删除及插入是很方便的,但存储量较大。对于含有 τ个非零元素的 n 阶方阵,所需存储单元为(7τ+ 2n)个。

3. 稀疏矩阵的优化排序方法

稀疏矩阵技术中,当采用高斯消去法或 LU 分解法求解时,往往会将系数矩阵中某些零元素变为非零元素,我们称由零变为非零的元素为“填入元素”,或简称“填入”、“填入元”。如果在求解过程中填入元的量太大,就会破坏矩阵的稀疏性,丧失稀疏矩阵技术的优势。

例 2.2.4　有一稀疏矩阵 T,矩阵 T 非零元素排列结构如下:

$$T^{0}=\begin{bmatrix}\times & \times & \times & \times\\ \times & \times & 0 & 0\\ \times & 0 & \times & 0\\ \times & 0 & 0 & \times\end{bmatrix}$$

以首行、首列非零元素(t_{11})为主元进行消元,T 阵变为

$$T^{1}=\begin{bmatrix}\underline{\times} & \times & \times & \times\\ 0 & \times & \acute{} & \acute{}\\ 0 & \acute{} & \times & \acute{}\\ 0 & \acute{} & \acute{} & \times\end{bmatrix}$$

其中“×”表示非零元素,“ ´ ”表示新填入的填入元,“$\underline{\times}$”表示主元。采用高斯消去法进行消元。第一次消去后,矩阵就变为满矩阵,原来的零元素全部变为填入,失去了稀疏性。其乘除运算量也与满矩阵相同,即

$$\text{乘除运算量}=\frac{n^3}{3}+n^2-\frac{n}{3}=36\ \text{次}$$

$$\text{填入量}=6$$

若将 T 矩阵的第一行与第四行对调,再将第一列与第四列对调,则 T 变为

$$T^0=\begin{matrix}\times & 0 & 0 & \times\\ 0 & \times & 0 & \times\\ 0 & 0 & \times & \times\\ \times & \times & \times & \times\end{matrix}$$

对此矩阵进行高斯消去运算，消去 t_{11} 后，T 为

$$T^1=\begin{matrix}\underline{\times} & 0 & 0 & \times\\ 0 & \times & 0 & \times\\ 0 & 0 & \times & \times\\ 0 & \times & \times & \times\end{matrix}$$

再以 t_{22}、t_{23} 为主元进一步消元后，T^1 变为 T^2 和 T^3，即

$$T^2=\begin{matrix}\times & 0 & 0 & \times\\ 0 & \underline{\times} & 0 & \times\\ 0 & 0 & \times & \times\\ 0 & 0 & \times & \times\end{matrix}\qquad T^3=\begin{matrix}\times & 0 & 0 & \times\\ 0 & \times & 0 & \times\\ 0 & 0 & \underline{\times} & \times\\ 0 & 0 & 0 & \times\end{matrix}$$

所用乘除运算量和填入量为：

乘除运算量= 10(包括回代运算量)

填入量= 0

也就是说矩阵经过行列调换之后，消去过程中没有产生填入，乘除运算量也减少了。

由此例可以看出，矩阵的排列次序对填入量和乘除运算量有较大影响。可以认为，对一个稀疏矩阵可能存在着一种最优的排列次序，使得在消元过程中产生的填入最少或总的乘除运算量最小。为了找到最优的行列次序，可以在所有非零元中挑选主元，但运算量太大(对于一个 n 阶满阵，行列排列方案达$(n!)^2$种)，因而这种全局优化方法是不现实的。目前尚无简单有效的全优方法。一般采用的是局部最优的方法。这里介绍两种实用的方法，即按填入量极小排列和按乘除运算量极小排列的方法。

(1) 局部填入量极小方法

这种方法称为 Tinney-Walker(简称 T-W)方法，它是按填入量极小选取主元的一种方法。它的要点是：

1) 在高斯消元或 LU 分解的每一步，选择此步矩阵中产生填入量最小的非零元为主元。主元的挑选范围，原则上可以是全部非零元，但工作量较大，一般以主对角线元素为对象。

2) 若同时有几个对角元素作为候选主元时所产生的填入数相同，则任选一个。这种任选只是为了在实用中减少运算量，不一定合理。因为这一步中产生的填入量虽然相同，对以后各步消元的影响却可能不同。

3) 假定矩阵是正定矩阵，或消元过程中的各阶子阵都具有对角优势。这样选主元时就可以只考虑其位置，而不必顾虑其数值。

T-W 法中需要计算填入的数目，故首先讨论填入产生的条件。

当我们采用高斯消去法对矩阵 T 作第一步消元时，即以 t_{11} 为主元，消去 t_{11} 以下全部非零元。为此需执行如下运算：

$$t_{1j}^{1} = t_{1j}^{0} / t_{11}^{0} \qquad (j = 2, \cdots, n) \tag{2.2.18}$$

$$t_{ij}^{1} = t_{ij}^{0} - t_{i1}^{0} \frac{t_{1j}^{0}}{t_{11}^{0}} \qquad \begin{matrix} i = 2, \cdots, n \\ j = 2, \cdots, n \end{matrix} \tag{2.2.19}$$

式(2.2.18)不会产生填入,则填入只可能由(2.2.19)式运算中产生。根据填入定义,从(2.2.19)式中可看出,当 t_{i1}^{0}, t_{1j}^{0} 不等于零,而 t_{ij}^{0} 等于零时,会产生填入 t_{ij}^{1}。可写成如下表达式:

$$t_{ij}^{0} = 0 \ \wedge \ t_{i1}^{0} \neq 0 \ \wedge \ t_{1j}^{0} \neq 0 \qquad \begin{matrix} i = 2, \cdots, n \\ j = 2, \cdots, n \end{matrix}$$

其中"∧"表示两个条件的"与"关系。

对于第 k 步消元产生填入的一般式为

$$t_{ij}^{k-1} = 0 \ \wedge \ t_{ik}^{k-1} \neq 0 \ \wedge \ t_{kj}^{k-1} \neq 0 \qquad \begin{matrix} i = k+1, \cdots, n \\ j = k+1, \cdots, n \end{matrix} \tag{2.2.20}$$

根据(2.2.20)式,采用一种几何划线的方法来判断填入的产生,则更加简便、直观和易于理解。下面以 T^{0} 中 t_{11} 为主元,说明几何划线法。观察

$$T^{0} = \begin{bmatrix} \times & \times & 0 & \times \\ \times & \times & \times & \times \\ 0 & 0 & \times & \times \\ \times & \times & 0 & \times \end{bmatrix}$$

中与主元同行和同列的非零元素。凡 $t_{i1} \neq 0$($i = 2, \cdots, 4$)的行作一族平行线,凡 $t_{1j} \neq 0$($j = 2, \cdots, 4$)的列做一族平行线,这两组平行线交点位置上的元素若满足(2.2.20)式的条件,则在此位置上产生填入,则 $t_{ij}^{1} \neq 0$。上式中 t_{24} 即为填入。

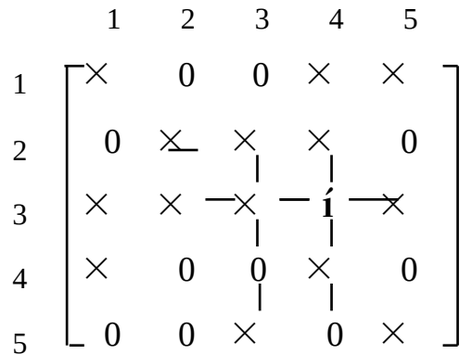
下面我们用几何划线法分析例 2.2.5,以此介绍 T-W 法的应用。

例 2.2.5　已知 T 矩阵中的非零元分布,用局部填入极小的方法寻找用高斯消去法求解过程中的主元次序。

在高斯消去过程的每一步,以对角元为候选主元,用划线法计算每一主元情况下的填入量。以 t_{11} 为主元进行消元时,所产生的填入如下面矩阵所示:

$$\begin{array}{c} \\ 1 \\ 2 \\ 3 \\ 4 \\ 5 \end{array} \begin{array}{c} \begin{array}{ccccc} 1 & 2 & 3 & 4 & 5 \end{array} \\ \begin{bmatrix} \times & 0 & 0 & \times & \times \\ 0 & \times & \times & \times & 0 \\ \times & \times & \times & \times & \times \\ \times & 0 & 0 & \times & \times \\ 0 & 0 & \times & 0 & \times \end{bmatrix} \end{array}$$

可以看出以 t_{11} 为候选主元,用划线法得到两个填入,即 t_{34} 和 t_{45}。填入量= 2。以 t_{22} 为候选主元时,产生填入为



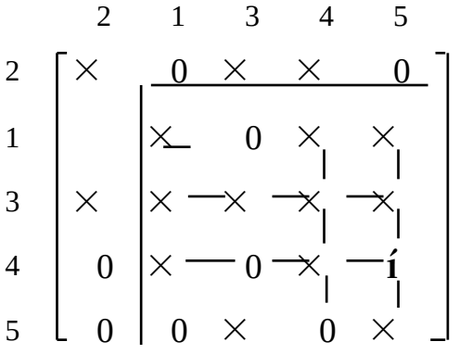
可以看出以 t_{22} 为候选主元，用划线法得到填入为 t_{34}，填入量= 1。

由此类推，可列出各候选主元下所产生的填入量，列于表 2.2.10 中。此时应选填入最

表 2.2.10　候选主元和填入量

候选主元	t_{11}	t_{22}	t_{33}	t_{44}	t_{55}
填入值	2	1	4	1	1

小者，现 t_{22}，t_{44}，t_{55} 产生填入数都为 1，则任选一个，确定 t_{22} 为主元。进行相应的行列交换



后，再在矩阵余下的 4 阶子阵中选主元。这时应将前面的填入作为非零元 t_{34} 加入到矩阵中。在余下的四阶子阵中选主元所产生的填入如表 2.2.11 所示。显然选 t_{44} 为主元，因它不产生填入。

表 2.2.11　候选主元与填入量

候选主元	t_{11}	t_{33}	t_{44}	t_{55}
填入值	1	2	0	1

如此继续选主元，最后确定的主元次序是 t_{22}，t_{44}，t_{11}，t_{33}，t_{55}。总的填入量为 1，总的乘除运算量为 12。

我们在实际求解时，一般先用 T-W 法做一次模拟消元，选出一种填入量极小的方案，并做相应的行列交换，将系数矩阵中非零元和填入的分布状况固定下来；以后求解就按照这种确定的主元顺序消元，以提高求解的效率。

应该指出，填入量极少不一定乘除运算量也极小，但有一定联系。

(2) 局部乘除运算量极小的排序方法

这种方法称为 Markowitz(简称 M) 方法，它是按乘除运算量极小来选取主元的一种方法。因为乘除法运算时间在计算机执行时间中占比例很大，因而尽量减少乘除运算次数是提高解题效率的一个重要途径。

乘除运算量极小排序方法的要点是:

1) 在高斯消元或 LU 分解的每一步, 选择矩阵中能使乘除运算量最少的对角主元作为主元。

2) 若同时有几个元素作候选主元时所需乘除运算量相同, 则任选一个。

3) 与前面 T-W 法一样, 也假定 T 是正定矩阵, 或 T 和在求解过程中的各子阵均为对角优势矩阵。这样, 在算法中可以不必考虑数值稳定性。

乘除运算次数的计算方法, 以高斯消去法表达式(2.2.3)为依据, 设选 t_{kk} 为主元, 则

乘除运算量= 第 k 行中 t_{kk} 以外的非零元数

+ 第 k 行中 t_{kk} 以外的非零元数× 第 k 列中 t_{kk} 以外的非零元数

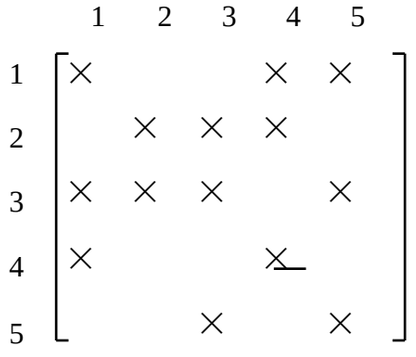
或写为

$$N_{ep} = (N_{rk} - 1)N_{ck} \tag{2.2.21}$$

其中 N_{ep} 为乘除运算次数, N_{rk} 为第 k 行中非零元数(包括 t_{kk}), N_{ck} 为第 k 列中的非零元数(包括 t_{kk})。

下面我们仍以前面例 2.2.5 中矩阵为例, 说明 Markowitz 方法。

例 2.2.6　已知 T 矩阵

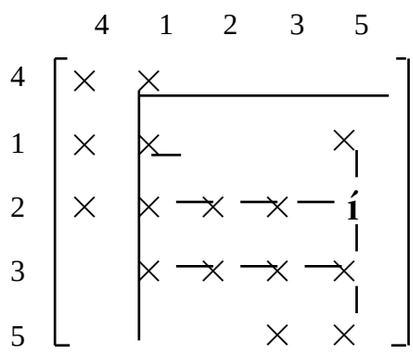


中的非零元分布, 采用乘除运算量极小方法寻找高斯求解过程中的 T 矩阵主元次序。

首先以 t_{11} 为主元, 乘除运算量= 2+ 2× 2= 6; 再以 t_{22} 为主元, 乘除运算量= 2+ 2× 1 = 4。类推出以各对角元为主元时的乘除运算量, 列于表 2.2.12 中。表中 t_{44}, t_{55} 为主元时乘除次数最少, 任选 t_{44} 为主元。这时要用前面所述方法计算以 t_{44} 为主元时的填入, 填入量 = 1, 即为 t_{21}。然后进行行列交换, 继续确定矩阵

表 2.2.12　候选主元和乘除量

候选主元	t_{11}	t_{22}	t_{33}	t_{44}	t_{55}
乘除次数	6	4	9	3	3



余下的子阵中主元的选取,其结果列入表 2.2.13 中。这时选 t_{11} 为主元,填入为 1,即 t_{25}。

表 2.2.13　候选主元和乘除量

候选主元	t_{11}	t_{22}	t_{33}	t_{55}
乘除次数	3	4	9	3

如此继续下去,最后确定的主元次序是 t_{44}, t_{11}, t_{22}, t_{33}, t_{55},总的填入数为 2,总的乘除运算量为 12。

从该例题中可以看出 Markowitz 方法的特点:

1) 在选主元时,只需计算乘除运算量,而不必寻找填入,比较简便。

2) 在每一步选定主元后,仍需找出填入的位置。

3) 该算法着眼于乘除运算量最少,但它仅是一种局部优化算法,故总的乘除运算量不一定比 T-W 法少。(本例中,采用 M 法与采用 T-W 法所需乘除量相等。)

由于稀疏矩阵结构不同,采用 T-W 法和 M 法会得到不同的结果,各有优劣,没有一定的规律。因此可以将两种方法结合起来应用。例如,先以乘除量极小方法选主元,当候选主元不唯一时,再采用填入量极小方法进一步确定。

4. 保证数值稳定性的方法

前面介绍的稀疏矩阵算法,是根据填入或乘除运算量极小的原则,改变原稀疏矩阵的非零结构,确定局部优化的主元排序。这种主元序列一旦确定,不论元素的数值如何变化,序列不再改变。因此对于方程的数值求解而言,实质上是不选主元的解法。我们在前面已经讲过,不选主元的高斯消去法可能会给数值稳定性造成很大威胁。除了前面介绍的采用双精度字长运算可以改善数值稳定性外,这里我们再介绍一些其它的处理方法。

由数值分析方法可知,为了保证数值稳定性,通常采用列主元、行主元或全主元法,但这些方法常会破坏矩阵的稀疏性,对于结构对称的矩阵还会破坏其对称性,往往几方面不能兼顾,至今还没有理想的最优方法。目前一般采取折衷的方法。

(1) 主元容限法

这种方法是对候选主元的绝对值加以限制,使其不得小于某规定限值 ε,在此主元容限范围内计算各主元的填入量或乘除运算量,加以比较,以确定主元。规定限制的方法可有许多种。

以列的主元容限法为例。当在消元的第 k 步时,从第 k 列非零元素 $t_{ik}^{k-1} \neq 0$ ($i = 1, 2, \cdots, n$) 中选取候选主元。设 N_Z, N_P 是如下定义的集合:

$$N_Z = \{ i \mid t_{ik}^{k-1} \neq 0 \quad (i = k, \cdots, n) \}$$

$$N_P = \{ i \mid |t_{ik}^{k-1}| \geq \varepsilon \quad (i \in N_Z) \}$$

N_P 集合中各元素均可作为候选主元。

ε 有各种取法,例如:

1) ε 为 N_Z 中各元素的平均值,即

$$\varepsilon = \frac{1}{n_i} \left| t_{ik}^{k-1} \right| \qquad (i = k, \cdots, n)$$

2) ε 等于 N_Z 中绝对值最大的元素乘以系数 α (α 为小于 1 的正数),亦即

$$\varepsilon = \alpha \max_i \left| t_{ik}^{k-1} \right| \qquad (i = k, \cdots, n)$$

3) 取 ε为

$$\varepsilon = \gamma_{min} + \beta(\gamma_{max} - \gamma_{min})$$

其中,

$$\gamma_{min} = \min_i \left| t_{ik}^{k-1} \right| \qquad (i = k, \cdots, n)$$

$$\gamma_{max} = \max_i \left| t_{ik}^{k-1} \right| \qquad (i = k, \cdots, n)$$

其中第 1) 种规定 ε的方法不必寻找最大、最小值, 比较方便。

同理, 也有相应的行主元容限法和全主元的容限法。

(2) 适时选主元方法

在非线性直流分析或瞬态分析过程中, 系数矩阵及其右端向量元素的数值都在变化。但除了信号有突变或步长太大情况下, 数值上应当具有某种连续性。因此可以考虑不必每次数值解都重新选主元。而是选一次主元, 形成一个非零结构以后, 若干次不变地使用同一个非零结构求解; 到适当的时候再选一次主元, 形成另一种非零结构, 这样可以提高求解效率。

2.2.4 复数方程组解法

在交流稳态分析中, 由于阻抗、电压、电流等均用复数表示, 所以电路方程是一个复数形式的线性代数方程组。常用的解法有两种。

第一种方法是直接求解复数方程组。采用与求解实系数方程组相同的求解方法和过程, 不同之处只在于每步运算采用复数运算而已。

求解实系数方程组的公式完全适用于求解复数方程组。下面以高斯消去法为例, 对复数运算作简单说明。

由式(2.2.3)式, 可知

$$t_{ij}^{k} = t_{ij}^{k-1} - t_{ik}^{k-1} \frac{t_{kj}^{k-1}}{t_{kk}^{k-1}} \tag{2.2.22}$$

设复数方程组 $TX = B$ 中, 复系数矩阵 T 的元素 t_{ij}表示为 $t_{ij} = t_{Rij}^{k} + jt_{Iij}$, 则式(2.2.22)的复数形式为

$$t_{Rij}^{k} + jt_{Iij}^{k} = t_{Rij}^{k-1} + jt_{Iij}^{k-1} - t_{Rik}^{k-1} - jt_{Iik}^{k-1} \frac{t_{Rkj}^{k-1} - jt_{Ikj}^{k-1}}{t_{Rkk}^{k-1} - jt_{Ikk}^{k-1}} \tag{2.2.23}$$

经化简, 得到

$$\begin{aligned} t_{Rij}^{k} &= t_{Rij}^{k-1} - \left(t_{Rik}^{k-1} E_{kj}^{k-1} - t_{Iik}^{k-1} F_{kj}^{k-1} \right) / M_{kk}^{k-1} \\ t_{Iij}^{k} &= t_{Iij}^{k-1} - \left(t_{Rik}^{k-1} F_{kj}^{k-1} + t_{Iik}^{k-1} E_{kj}^{k-1} \right) / M_{kk}^{k-1} \end{aligned} \tag{2.2.24}$$

其中

$$\begin{aligned} E_{kj}^{k-1} &= t_{Rkj}^{k-1} t_{Rkk}^{k-1} + t_{Ikj}^{k-1} t_{Ikk}^{k-1} \\ F_{kj}^{k-1} &= t_{Ikj}^{k-1} t_{Rkk}^{k-1} - t_{Rkj}^{k-1} t_{Ikk}^{k-1} \\ M_{kk}^{k-1} &= t_{Rkk}^{k-1\,2} + t_{Ikk}^{k-1\,2} \end{aligned} \tag{2.2.25}$$

以上各式中, $k = 1, 2\cdots, n$; $i = k+1, \cdots, n$; $j = k+1, \cdots, n$。其余运算均可类似地推导。

如果采用的计算机语言中有复数运算功能, 只要在程序开始时先将有关数组和变量定义为复数型, 即可将实系数的解方程算法转变为等价的复数方程组求解算法。

第二种方法是将复数方程组等价为实系数方程组求解。

将复数方程组 $TX = B$ 写为

$$(T_R + jT_I)(X_R + jX_I) = B_R + jB_I \tag{2.2.26}$$

上式两端的实部和虚部应分别相等，即得到等价实数方程组

$$\begin{aligned} T_R X_R - T_I X_I &= B_R \\ T_I X_R + T_R X_I &= B_I \end{aligned} \tag{2.2.27}$$

写成矩阵形式为

$$TX = \begin{matrix} T_R & -T_I \\ T_I & T_R \end{matrix} \begin{matrix} X_R \\ X_I \end{matrix} = \begin{matrix} B_R \\ B_I \end{matrix} \tag{2.2.28}$$

若原复数方程组为 n 阶，则(2.2.28)式表示的等价实数方程组为 2n 阶，存储量增加一倍，计算量也显著增加。我们知道，在交流稳态分析之前必须先求解直流工作点。求直流工作点所形成的实系数矩阵 T 与交流稳态分析的系数矩阵 T，不仅阶数不同，非零结构不相同，求解时所需的存储和指针系统也不相同，因此存储量和计算量会成倍增加。

习　　题

2.1　写出题图 2.1 所示各电网络的关联矩阵，并用它建立表矩阵方程。

题图　2.1

2.2　列出题图 2.2 所示网络的 KCL、KVL 和支路特性方程，并由此形成网络的节点方程。

题图　2.2

2.3　用元件送值表直接列出题图 2.3 所示各网络的节点方程。

题图　2.3

2.4　用改进节点法列出题图 2.4 所示各电网络的混合方程组。

题图　2.4

2.5　用改进节点法列出题图 2.5 所示电网络的混合方程组。

题图　2.5

2.6　用高斯消去法解下列矩阵方程:

(1)
$$\begin{bmatrix} 4 & 6 \\ 2 & 8 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} = \begin{bmatrix} 2 \\ 0 \end{bmatrix}$$

(2)
$$\begin{bmatrix} 2 & 4 & 6 \\ 6 & 4 & 2 \\ 1 & 8 & 4 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} = \begin{bmatrix} 4 \\ 12 \\ 13 \end{bmatrix}$$

2.7 求出题 2.6 中矩阵方程的 LU 分解,并用向前-向后代换求解。

2.8 用 LU 分解法求下列各矩阵方程:

(1)
$$\begin{bmatrix} 1 & 0 & 0 & 2 & 1 \\ 3 & 2 & 0 & 0 & 2 \\ 0 & 1 & 1 & 0 & 0 \\ 0 & 0 & 1 & 2 & 1 \\ 0 & 0 & 1 & 1 & 1 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \\ x_5 \end{bmatrix} = \begin{bmatrix} -1 \\ 3 \\ 0 \\ 0 \\ 1 \end{bmatrix}$$

(2)
$$\begin{bmatrix} 2 & 4 & 0 & 2 & 0 \\ 0 & 3 & 6 & 0 & 3 \\ 1 & 2 & 1 & 2 & 0 \\ 2 & 4 & 1 & 4 & 0 \\ 0 & 1 & 2 & 2 & 3 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \\ x_5 \end{bmatrix} = \begin{bmatrix} 4 \\ 0 \\ 3 \\ 6 \\ 2 \end{bmatrix}$$

(3)
$$\begin{bmatrix} 11 & -10 & 0 & 0 \\ -10 & 19 & -4 & 0 \\ 0 & -4 & 39 & -20 \\ 0 & 0 & -20 & 22 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \end{bmatrix} = \begin{bmatrix} 10 \\ 20 \\ -20 \\ -15 \end{bmatrix}$$

(4)
$$\begin{bmatrix} 3 & -1 & -1 & 0 \\ -1 & 3 & -1 & -1 \\ -1 & -1 & 3 & -1 \\ 0 & -1 & -1 & 3 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \end{bmatrix} = \begin{bmatrix} 5 \\ -10 \\ 10 \\ -5 \end{bmatrix}$$

(5)
$$\begin{bmatrix} 1 & 0 & 1 & 0 \\ 2 & 2 & 2 & 4 \\ 0 & 1 & 3 & 11 \\ 1 & 0 & -1 & -5 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ 0 \\ 3 \end{bmatrix}$$

2.9 已知电路方程的系数矩阵的非零元分布,即有

$$T_1 = \begin{bmatrix} t_{11} & 0 & 0 & t_{14} & 0 & t_{16} \\ 0 & t_{22} & 0 & 0 & t_{25} & 0 \\ 0 & 0 & t_{33} & t_{34} & 0 & 0 \\ 0 & t_{42} & 0 & t_{44} & 0 & t_{46} \\ 0 & 0 & t_{53} & 0 & t_{55} & 0 \\ t_{61} & 0 & 0 & t_{64} & 0 & t_{66} \end{bmatrix} \qquad T_2 = \begin{bmatrix} t_{11} & 0 & t_{13} & 0 & 0 \\ 0 & t_{22} & 0 & 0 & t_{25} \\ t_{31} & 0 & t_{33} & 0 & 0 \\ 0 & t_{42} & 0 & t_{44} & 0 \\ 0 & 0 & 0 & t_{54} & t_{55} \end{bmatrix}$$

(1) 请列出采用线性表存储法和单链表存储法的存储方案。

(2) 请列出采用局部填入量极小方法的送主元次序和乘除运算量、填入量。

(3) 请列出采用局部乘除量极小方法的选主元次序和乘除运算量、填入量。

2.10　对题 2.3(1)、(2)和题 2.4(1)、(2)所列出的电路方程系数矩阵,

(1) 采用局部填入量极小算法列出选主元的次序和乘除运算量、填入量。

(2) 采用局部乘除量极小算法列出选主元的次序和乘除运算量、填入量。

第 3 章　半导体器件模型

在第 2 章中,我们已经介绍了一些简单的电子元件模型,如电阻、电感、独立源和四种受控源等,它们的支路特性方程是线性的,因此通常称为线性元件。当然,如果这些元件的支路特性方程是非线性的,也可以作为非线性元件,如非线性电阻、非线性电容等等。半导体器件是目前构成电子线路,尤其是集成电路最基本的器件,它们的支路特性方程是非线性的,通常称为非线性器件。用电子计算机进行电路分析,很重要的问题就是如何用等效的数学模型来描述这些元器件。对于一个较理想的器件模型,一方面要求它能正确地反映元器件的电学特性,参数意义简明,易于提取,另外又必须便于在计算机上计算。实际上,在计算方法正确和计算机字长足够的前提下,电路分析结果的正确性主要由元器件模型的正确性和精度决定。

一般而言,有两种建立器件模型的方法,一种是建立在器件物理原理基础上的模型,另一种是根据输入、输出外特性来构成的模型。前者必须知道器件的内部工作原理,其模型参数与物理机理有密切的关系,因此参数的适应范围较大,但参数的测定和计算通常比较麻烦。后者要了解电路的工作原理,但不必了解具体器件的内部机理,模型参数可通过直接测量而获得,缺点是模型参数适用的工作范围窄,并且与测试条件有关。

在电路 CAD 中,半导体器件的模型具有较为复杂的特性,例如双极型晶体管和 MOS 场效应晶体管的模型参数有 40 个之多,因此在这一章中我们用较多的篇幅介绍半导体器件模型。半导体器件模型大都是以第一种建模方式,即从器件的物理机理为基础的方程出发构造的模型。我们着重介绍的是以 SPICE 程序中所用的半导体模型的基本型。针对不同的分析要求,比如直流稳态分析、频域分析、时域分析等等,需要由这些基本型模型形成不同的等效模型,我们将在以后各章中陆续介绍。

近年来,集成电路得到广泛的应用。随着集成电路规模的增大和功能复杂程度的增加,如果对集成电路中每个晶体管都采用相应的模型,那不仅需要相当大的计算机内存,而且会耗费较多的机时。采用反映集成电路端口特性的宏模型是一种非常有效的手段,我们将介绍宏模型的一些基本知识。

3.1　二极管模型

在 SPICE 程序中,结型(或肖特基)二极管的模型如图 3.1.1 所示。图中 R_s 是二极管的材料电阻,称为欧姆电阻;C_D 是由电荷存储效应而引起的等效电容;I_D 是非线性电流源。

非线性电流源 I_D 与加在它两端的电压 U_D 之间的关系式如下:

$$I_D = f(U_D)$$

$$= \begin{cases} I_S\left(e^{qU_D/nkT} - 1\right) + U_D G_{min} & -5\dfrac{nkT}{q} \le U_D \\ -I_S + U_D G_{min} & -BV < U_D < -5\dfrac{nkT}{q} \\ -I_{BV} & U_D = -BV \\ -I_S\left(e^{-q(BV+U_D)/kT} - 1 + \dfrac{qBV}{kT}\right) & U_D < -BV \end{cases} \tag{3.1.1}$$

其中，I_S——饱和电流(A)；

q——电子电荷(1.602×10^{-19}C)；

k——玻尔兹曼常数(1.38×10^{-23}J/K)；

T——绝对温度(K)；

n——发射系数(硅管 1.2～2.0)；

BV——反向击穿电压(V)；

I_{BV}——反向击穿时的电流(A)。

图 3.1.2 中给出了式(3.1.1)所描述的四段特性。为了有助于非线性支路的收敛，SPICE 程序在二极管的 PN 结上并了一个小电导 G_{min}，它的隐含值是 10^{-12}S。G_{min}的具体作用我们将在第 4 章 4.6 节中作进一步介绍，一般情况下 G_{min}的存在不会影响二极管的正常特性。

图 3.1.1　二极管符号和模型

图 3.1.2　实际二极管伏安特性

二极管的电荷存储效应包括两部分，一部分是在 PN 结势垒电容上存储的电荷，它等于势垒电容对 PN 结电压的积分；另一部分是注入少数载流子形成的电荷存储(对肖特基二极管，没有此分量)，它和正向电流成正比。总的电荷存储量 Q_D 为

$$Q_D = \begin{cases} \tau I_D + C_{J0}\int_0^{U_D}\left(1-\dfrac{U}{\phi}\right)^{-m} dU & U_D < FC\times\phi \\ \tau I_D + C_{J0}F_1 + \dfrac{C_{J0}}{F_2}\int_{FC\times\phi_D}^{U_D}\left(F_3 + \dfrac{mU}{\phi}\right) dU & U_D \ge FC\times\phi \end{cases} \tag{3.1.2}$$

也可以用等效电容来表示上式，即有

$$C_D = \frac{dQ_D}{dU_D} = \begin{cases} \tau \frac{dI_D}{dU_D} + C_{J0}\left(1 - \frac{U_D}{\phi}\right)^{-m} & U_D < FC \times \phi \\ \tau \frac{dI_D}{dU_D} + \frac{C_{J0}}{F_2}\left(F_3 + \frac{mU_D}{\phi}\right) & U_D \geq FC \times \phi \end{cases} \tag{3.1.3}$$

其中

$$\frac{dI_D}{dU_D} = \frac{qI_S}{\eta kT}\exp\left(\frac{qU_D}{\eta kT}\right) \tag{3.1.4}$$

$$\begin{aligned} F_1 &= \frac{\phi}{1-m}\left[1-(1-FC)^{1-m}\right] \\ F_2 &= (1-FC)^{1+m} \\ F_3 &= 1-FC(1+m) \end{aligned} \tag{3.1.5}$$

此外，τ——少数载流子的渡越时间；

C_{J0}——零偏置时 PN 结的耗尽层电容；

ϕ——PN 结自建势，对结型二极管的典型值是 0.7～0.8V；

m——电容梯度因子，典型值是 0.3～0.5；

FC——正偏耗尽电容公式的系数，典型值是 0.5。

二极管等效电容 C_D 是由两部分电容组成的：一个是少数载流子注入的电荷存储产生的扩散电容 C_s，另一个是由 PN 结耗尽层电荷存储产生的耗尽层电容 C_d，即

$$C_D = C_s + C_d \tag{3.1.6}$$

其中，

$$C_s = \tau \frac{dI_D}{dU_D} \tag{3.1.7}$$

$$C_d = \begin{cases} C_{J0}\left(1 - \frac{U_D}{\phi}\right)^{-m} & U_D < FC \times \phi \\ \frac{C_{J0}}{F_2}\left(F_3 + \frac{mU_D}{\phi}\right) & U_D \geq FC \times \phi \end{cases} \tag{3.1.8}$$

图 3.1.3 中所示的是 PN 结耗尽层电容 C_d 随电压 U_D 变化的特性，图中 FC= 0.5。

图 3.1.3　耗尽层电容 C_d 随 U_D 变化特性

SPICE 程序中二极管模型共有 14 个模型参数，它们的符号、在 SPICE 中的关键字、名称和隐含值等如表 3.1.1 所示。

表 3.1.1　SPICE 二极管模型参数表

序号	符号	SPICE 关键字	名称	隐含值	单位	举例
1	I_S	IS	饱和电流	10^{-14}	A	2×10^{-15}A
2	R_s	RS	欧姆电阻	0	Ω	10Ω
3	n	N	发射系数	1		1.2
4	τ	TT	渡越时间	0	s	1ns
5	C_{Jo}	CJO	零偏置结电容	0	F	2pF
6	ϕ	VJ	结电压	1	V	0.6V
7	m	M	电容梯度因子	0.5		0.33
8	E_g	EG	禁带宽度	1.11	eV	1.11eV(硅) 0.69eV(锑) 0.67eV(锗)
9	p_t	XTI	饱和电流温度系数	3.0		3.0
10	FC	FC	正偏耗尽电容公式系数	0.5		0.5
11	BV	BV	反向击穿电压	∞	V	40V
12	I_{BV}	IBV	反向击穿时电流	10^{-3}	A	10^{-3}A
13	K_f	KF	闪烁噪声系数	0		
14	a_f	AF	闪烁噪声指数	1		

前述的各个公式都是在常温下计算的,SPICE 程序中常温是 27℃(300K)。然而,半导体器件中不少参数是温度的函数,为了反映参数随环境温度的变化,SPICE 程序给出了一些参数,如 I_S, ϕ, C_D 等的温度修正公式。I_S 的温度修正公式为

$$I_S(T_2) = I_S(T_1)\left(\frac{T_2}{T_1}\right)^{p_t/n}\exp\left(-\frac{qE_g(300)}{kT_2}\left(1-\frac{T_2}{T_1}\right)\right) \qquad (3.1.9)$$

对 PN 结管 $p_t=3$,对肖特基管 $p_t=2$。对硅二极管 $E_g(300)=1.11$eV。ϕ 与 C_D 的温度修正公式分别为

$$\phi(T_2) = \frac{T_2}{T_1}\phi(T_1) - 2\frac{kT_2}{q}\ln\left(\frac{T_2}{T_1}\right)^{1.5} - \frac{T_2}{T_1}E_g(T_1) - E_g(T_2) \qquad (3.1.10)$$

$$C_D(T_2) = C_D(T_1)\left[1+m\left(400\times10^{-6}(T_2-T_1)-\frac{\phi(T_2)-\phi(T_1)}{\phi(T_1)}\right)\right] \qquad (3.1.11)$$

式中,

$$E_g(T) = E_g(0) - \frac{\alpha T^2}{\beta+T} \qquad (3.1.12)$$

对于硅器件,$\alpha=7.02\times10^{-4}$,$\beta=1108$,$E_g(0)=1.16$eV。

SPICE 程序中用.MODEL 语句描述器件模型参数,用.TEMP 语句描述温度条件。

例如,一个典型的模型名为 MD 的二极管模型描述语句:

```
.MODEL  MD  D  IS= 1E- 13  RS= 10  TT= 10ns
+ CJO= 2pF  VJ= 0.65  M= 0.35
```

3.2 双极型晶体管模型

双极型晶体管(简称 BJT)模型种类很多,但是在电路 CAD 领域中使用得最为普遍的是 Ebers-Moll 模型(简称 EM 模型)和 Gummel-Poon 模型(简称 GP 模型)。EM 模型本身是一种大信号非线性直流模型,不考虑电荷存储效应和二阶效应。但目前原始的 EM 模型经过了不断的改进,已成为包括各种效应的较为完善的模型。GP 模型包括了主要的二阶效应,是一种数学推导上更加严格和完整的模型。本节中我们着重介绍 EM 模型和它的各种改进模型。

3.2.1 EM1 模型

Ebers 和 Moll 于 1954 年提出的 Ebers-Moll 模型基本上是一种简单的非线性直流模型,目前又称 EM1 模型。EM1 模型不考虑器件中的电荷储存特性。它适用于所有工作区域,即饱和区、反向工作区、正向工作区和截止区。实际上,所有的直流和大信号非线性模型都是以 EM1 模型为基础。

EM1 模型主要有三种形式:注入型模型、传输型模型和混合 π模型。我们以 NPN 型晶体管为例,介绍这三种模型。

1. 注入型模型

假定在一维小注入条件下,忽略晶体管基区宽度随 U_{BC} 的变化,则晶体管的电流电压特性可由 Ebers-Moll 方程来表示,即

$$I_E = -I_{ES}\left(\exp\frac{qU_{BE}}{kT}-1\right)+\alpha I_{CS}\left(\exp\frac{qU_{BC}}{kT}-1\right) \tag{3.2.1}$$

$$I_C = \alpha I_{ES}\left(\exp\frac{qU_{BE}}{kT}-1\right)-I_{CS}\left(\exp\frac{qU_{BC}}{kT}-1\right) \tag{3.2.2}$$

其中,I_E ——发射极电流;

I_C ——收集极电流;

U_{BE} ——发射结上所加电压;

U_{BC} ——收集结上所加电压;

I_{ES} ——收集结电压为零时发射结反向饱和电流;

I_{CS} —— 发射结电压为零时收集结反向饱和电流;

α ——正向共基极电流放大系数;

α ——反向共基极电流放大系数。

令

$$I_F = I_{ES}\left(\exp\frac{qU_{BE}}{kT}-1\right) \tag{3.2.3}$$

$$I_R = I_{CS}\left(\exp\frac{qU_{BC}}{kT}-1\right) \tag{3.2.4}$$

则(3.2.1)式和(3.2.2)式可写为

$$I_E = -I_F + \alpha I_R \tag{3.2.5}$$

$$I_C = \alpha I_F - I_R \tag{3.2.6}$$

由(3.2.5)式和(3.2.6)式,可得到图 3.2.1 所示的 EM1 注入型模型。之所以称为注入型模型,是因为参考电流 I_F 和 I_R 分别为发射结和集电结正向偏置时的正向注入电流。在一定温度下,描述注入型模型需要四个参数,即 I_{ES}, I_{CS}, α 和 α。若利用互易定理,可得

$$\alpha I_{ES} = \alpha I_{CS} = I_S \tag{3.2.7}$$

其中 I_S 称为晶体管饱和电流。从而模型参数可减少一个。若采用共发射极接法,其电流增益 β 和 β 与 α 和 α 的关系式为

$$\beta_F = \frac{\alpha}{1-\alpha}, \qquad \beta_R = \frac{\alpha}{1-\alpha} \tag{3.2.8}$$

图 3.2.1 注入型 EM1 模型

因此,只用 I_S, β 和 β_R 三个模型参数就可以确定 EM1 注入型模型。

2. 传输型模型

传输型模型与注入型模型的差别仅仅在所选用的参考电流不同。在传输型模型中,参考电流是指流经模型电流源的传输电流 I_{FC} 和 I_{RC},它们可以写成

$$I_{FC} \approx \alpha I_F = I_S \exp\frac{qU_{BE}}{kT} - 1 \tag{3.2.9}$$

$$I_{RC} \approx \alpha I_R = I_S \exp\frac{qU_{BC}}{kT} - 1 \tag{3.2.10}$$

于是可得如图 3.2.2 所示的传输型 EM1 模型。

用传输电流 I_{FC} 和 I_{RC} 来描述晶体管端电流,其关系式为

$$I_C = I_{FC} - \frac{I_{RC}}{\alpha} \tag{3.2.11}$$

$$I_B = \left(\frac{1}{\alpha} - 1\right) I_{FC} + \left(\frac{1}{\alpha} - 1\right) I_{RC} \tag{3.2.12}$$

$$I_E = -\frac{1}{\alpha} I_{FC} + I_{RC} \tag{3.2.13}$$

传输型 EM1 模型需要三个参数,即 I_S, α 和 α。

从数学上看,传输型模型和注入型模型是等价的,差别仅在于表示符号不同而不是模型形式不同。但从实验观点看,传输型模型两个参考电流随结电压变化的曲线在好几个数量级范围内重合且保持线性,这是传输型模型优于注入型模型

图 3.2.2 传输型 EM1 模型

的关键所在。

3. 混合 π模型

这是传输型模型的一种变型。在传输型模型中,用收集极和发射极之间单一电流源代替两个等效电流源,即可得到非线性混合 π模型,如图 3.2.3 所示。

混合 π模型中的电流源电流 I_{CT} 为

$$I_{CT} = I_{FC} - I_{RC} = I_S \exp\frac{qU_{BE}}{kT} - \exp\frac{qU_{BC}}{kT} \tag{3.2.14}$$

晶体管端电流可以表示为

$$I_E = -\frac{I_{FC}}{\beta_F} - I_{CT} \tag{3.2.15}$$

$$I_B = \frac{I_{FC}}{\beta_F} + \frac{I_{RC}}{\beta_R} \tag{3.2.16}$$

$$I_C = I_{CT} - \frac{I_{RC}}{\beta_R} \tag{3.2.17}$$

图 3.2.3 混合 π型EM1 模型

在 SPICE 程序中双极型晶体管混合 π模型的端电流分为四个区来描述,即正向工作区、反向区、饱和区和截止区。

(1) 正向工作区: $U_{BE} > -5\frac{kT}{q}$ 且 $U_{BC} \leq -5\frac{kT}{q}$

$$I_C = I_S\left(e^{qU_{BE}/kT} + \frac{1}{\beta_R}\right) + U_{BE}\left(-1 + \frac{1}{\beta_R}\right) U_{BC} G_{min} \tag{3.2.18}$$

$$I_B = I_S\left(\frac{1}{\beta_F} e^{qU_{BE}/kT} - 1 - \frac{1}{\beta_R}\right) + \frac{U_{BE}}{\beta_F} + \frac{U_{BC}}{\beta_R} G_{min} \tag{3.2.19}$$

(2) 反向区: $U_{BE} \leq -5\frac{kT}{q}$ 且 $U_{BC} > -5\frac{kT}{q}$

$$I_C = -I_S\left(e^{qU_{BC}/kT} + \frac{1}{\beta_R}\right) e^{qU_{BC}/kT} - 1 + U_{BE}\left(-1 + \frac{1}{\beta_R}\right) U_{BC} G_{min} \tag{3.2.20}$$

$$I_B = -I_S\left(\frac{1}{\beta_F} - \frac{1}{\beta_R}\right) e^{qU_{BC}/kT} - 1 + \frac{U_{BE}}{\beta_F} + \frac{U_{BC}}{\beta_R} G_{min} \tag{3.2.21}$$

(3) 饱和区: $U_{BE} > -5\frac{kT}{q}$ 且 $U_{BC} > -5\frac{kT}{q}$

$$I_C = I_S\left(e^{qU_{BE}/kT} - e^{qU_{BC}/kT} - \frac{1}{\beta_R} e^{qU_{BC}/kT} - 1\right) + U_{BE}\left(-1 + \frac{1}{\beta_R}\right) U_{BC} G_{min} \tag{3.2.22}$$

$$I_B = I_S\left(\frac{1}{\beta_F} e^{qU_{BE}/kT} - 1 + \frac{1}{\beta_R} e^{qU_{BC}/kT} - 1\right) + \frac{U_{BE}}{\beta_F} + \frac{U_{BC}}{\beta_R} G_{min} \tag{3.2.23}$$

(4) 截止区: $U_{BE} \leq -5\frac{kT}{q}$ 且 $U_{BC} \leq -5\frac{kT}{q}$

$$I_C = \frac{I_S}{\beta_R} + U_{BE} - \left(1 + \frac{1}{\beta_R}\right) U_{BC} G_{min} \tag{3.2.24}$$

$$I_B = -I_S \frac{\beta_F + \beta_R}{\beta_F \beta_R} + \frac{U_{BE}}{\beta_F} + \frac{U_{BC}}{\beta_R} G_{min} \tag{3.2.25}$$

如果是 PNP 管, 则电压的极性和电流方向都应做相应的变化。

混合 πEM1 模型包含三个模型参数: I_S, β_F 和 β_R。它的优点是, 在交流小信号分析中很容易将混合 π模型转变为小信号线性混合 π模型。但是混合 π模型中的两个二极管已经不再代表发射结和收集结二极管了, 只表示晶体管基极电流中的两个分量而已。

EM1 模型形式上非常简单, 但仍是一种十分精确的直流非线性模型。EM1 模型的局限性主要在于忽略了晶体管的电荷存储效应和各端上的欧姆电阻。这些效应将在下面 EM2 模型中予以考虑。

3.2.2 EM2 模型

EM1 模型只能模拟晶体管的直流特性, 为了更精确地模拟晶体管的频域和时域特性, 在 EM1 模型的基础上增加八个新元件, 构成 EM2 模型。完整的 EM2 模型如图 3.2.4 所示。它是在非线性混合 π模型上加了三个欧姆电阻 $R_{CC'}$, $R_{EE'}$ 和 $R_{BB'}$, 两个扩散电容 C_{DE} 和 C_{DC} 以及两个结电容 C_{JE}, C_{JC} 和一个衬底电容 C_{SUB}。

图 3.2.4 EM2 模型

EM2 模型对电荷存储效应提供了一个一阶模型, 从而实现了有限的频率与时间响应。

1. 串联电阻的影响

在实际 BJT 晶体管中, 发射极、基极和收集极均有一定的串联电阻, 这些串联电阻对晶体管的瞬态特性、频率特性以至于直流特性都有一定的影响。在引入这些串联电阻后, 产生三个内节点, 我们称为 C′, B′ 和 E′。

(1) 收集极串联电阻 $R_{CC'}$

串联电阻 $R_{CC'}$ 对晶体管在饱和区的特性曲线及饱和压降有很大影响。在低的收集极-

发射极电压时，R_{CC}'将使饱合区中的曲线斜率减小，如图 3.2.5 所示。

图 3.2.5 R_{CC}'对 I_C 随 U_{CE} 变化特性的影响

图 3.2.6 EM2 模型中 R_{BB}'和 R_{EE}'对 $\ln I_C$ 和 $\ln I_B$ 随 U_{BE}变化特性的影响

在 EM2 模型中 R_{CC}'是个常量，但在一个实际晶体管中它是收集极电流和基极-收集极电压的函数。各种晶体管的 R_{CC}'可能有很大差别，从分立器件时的几欧姆到标准集成器件的几百欧姆。

(2) 基极串联电阻 R_{BB}'

基极串联电阻 R_{BB}'是个重要的模型参数。通常，它对小信号和暂态响应的影响最大。另外，由于 R_{BB}'与工作点有密切关系，它也是最难精确测定的参数之一。在 EM2 模型中 R_{BB}'是常量，它的典型值可以从几欧姆(高频器件)至几千欧姆(低频器件)。由 EM2 模型中测得的 $\ln I_C$ 和 $\ln I_B$ 随 U_{BE}变化曲线，可以看出 R_{BB}'的直流效应，如图 3.2.6 所示。

(3) 发射极串联电阻 R_{EE}'

发射极串联电阻 R_{EE}'一般很小，大约 1 欧姆数量级，对小功率管可以忽略不计。R_{EE}'的主要影响是使发射结电压减小 $R_{EE}'I_E$，并影响收集极电流和基极电流，如图 3.2.6 所示。

2. 电荷存储效应

EM2 模型中，电荷存储效应通过引入三类电容进行模拟。它们是：两个非线性结电容、两个非线性扩散电容和一个恒定的衬底电容。

(1) 结电容 C_{JE}，C_{JC}

结电容是当外加电压变化时，耗尽层电荷随之变化所引起的电容效应，也称为势垒电容。NPN 晶体管的发射结电容随发射结电压变化的表达式为

$$C_{JE}=\begin{cases} C_{JEo}\left(1-\dfrac{U_{BE}'}{\phi}\right)^{-m_E} & U_{BE}<FC\times\phi \\ \dfrac{C_{JEo}}{F_2}\left(F_3+\dfrac{m_E U_{BE}}{\phi}\right) & U_{BE}\geq FC\times\phi \end{cases} \tag{3.2.26}$$

其中，

$$F_2=(1-FC)^{1+m_E} \tag{3.2.27}$$

$$F_3=1-FC(1+m_E)$$

C_{JEo}——$U_{B'E'}=0$ 时发射结电容值;

ϕ—— 发射区-基区内建势垒电位。典型值 0.7～0.8V;

m_E—— 发射区-基区电容梯度因子。典型值是 0.333～0.5。

集电结电容随结电压变化表达式为

$$C_{JC}=\begin{cases} C_{JCo}\left(1-\dfrac{U_{BC}}{\phi}\right)^{-m_C} \\ \dfrac{C_{JCo}}{F_2}\left(F_3+\dfrac{m_C U_{BC}}{\phi}\right) \end{cases} \qquad \begin{matrix} \\ U_{BC}\geq FC\times\phi \end{matrix} \tag{3.2.28}$$

其中,

$$\begin{aligned} F_2&=(1-FC)^{1+m_C} \\ F_3&=1-FC(1+m_C) \end{aligned} \tag{3.2.29}$$

C_{JCo}——$U_{B'C'}=0$ 时集电结电容值;

ϕ—— 集电区-基区内建势垒电位。典型值 0.7～0.8V;

m_C—— 集电区-基区电容梯度因子。典型值 0.333～0.5。

为了避免当结电压 $U\to\phi$ 时,产生无限大的结电容,在 $U>\phi/2$ 时,结电容公式如下:

$$\begin{aligned} C_{JE}&=2^{m_E}C_{JEo}(1-m_E)+2\frac{m_E}{\phi}U_{B'E'} \\ C_{JC}&=2^{m_C}C_{JCo}(1-m_C)+2\frac{m_C}{\phi}U_{B'C'} \end{aligned} \tag{3.2.30}$$

(2) 扩散电容 C_{DE}, C_{DC}

扩散电容模拟了晶体管内少数载流子注入所引起的电荷存储效应。这种存储电荷由两部分组成:正向存储电荷 Q_{DE} 和反向存储电荷 Q_{DC}。

正向存储电荷和传输电流 I_{FC} 成正比,即

$$Q_{DE}\propto\tau_F I_{FC} \tag{3.2.31}$$

其中 τ_F 为正向渡越时间。反向存储电荷与传输电流 I_{RC} 成正比,即

$$Q_{DC}\propto\tau_R I_{RC} \tag{3.2.32}$$

其中 τ_R 为反向渡越时间。与这两种电荷对应的两个扩散电容定义为

$$C_{DE}=\frac{Q_{DE}}{U_{B'E'}}\approx\tau_F\frac{qI_{FC}}{kT} \tag{3.2.33}$$

$$C_{DC}=\frac{Q_{DC}}{U_{B'C'}}\approx\tau_R\frac{qI_{RC}}{kT} \tag{3.2.34}$$

正向渡越时间 τ_F 可以由公式

$$\tau_F=\frac{1}{2\pi f_T}-C_{JC}R_{CC'} \tag{3.2.35}$$

计算式中 f_T 为晶体管特征频率。

(3) 衬底电容 C_{SUB}

在集成电路中,衬底电容很重要。C_{SUB} 的典型值为 1～2pF。

完整的 EM2 模型是在 EM1 模型上增加八个元件而实现的,这八个元件是 $R_{CC'}$, $R_{BB'}$, $R_{EE'}$, C_{JE}, C_{JC}, C_{DE}, C_{DC}, C_{SUB}。为了表示这八个元件的特性,一共需要增加十二个模型参数,

它们是: $R_{CC'}$, $R_{BB'}$, $R_{EE'}$, C_{JE0}, ϕ, m_E, C_{JC0}, ϕ, m_C, τ, τ, C_{SUB}。原 EM1 模型需有三个模型参数 I_S, β 和 β, 因此, EM2 模型共需十五个模型参数。

EM2 模型的适用面较广, 特别适用于数字电路, 是目前电路模拟中使用最多、且最广的一种模型。我们在以后章节中所涉及的 BJT 晶体管, 都以 EM2 模型为准。然而, EM2 模型也存在一些局限性, 比如没有考虑基区宽度调制以及 β随电流变化的效应等等。这些效应将在 EM3 和 GP 模型中予以考虑。

3.2.3 EM3 模型

晶体管中存在着许多二阶效应, 比如基区宽度调制效应、大注入效应、小电流效应等等, 这些效应在某种条件下对晶体管特性有一定的影响。如果说 EM1 是一种简单的直流模型, EM2 是包含对电荷存储效应和欧姆电阻进行一阶模拟的模型, 那么 EM3 模型则是在 EM2 模型的直流特性、电荷存储效应以及温度特性等方面进行了二阶模拟的模型。EM3 模型增加了以下特点:

1) 反映了基区宽度调制效应;
2) 反映了 β随电流和电压的变化;
3) 使跨接于 $R_{BB'}$ 的集电极-基极电容效应更符合实际的考虑;
4) τ 随电流而变化;
5) 器件参数随温度而变化。

这些效应主要是通过数学公式来描述的。如图 3.2.7 所示 EM3 模型只增加了两个二极管(D_1, D_2)和一个结电容(C_{JC}), 仅仅反映了 β随电流变化的部分特性。所以 EM3 基本上是个数学模型。

图 3.2.7 局部 EM3 模型

下面介绍 EM3 模型的几种主要的二阶效应。

1. 基区宽度调制效应

基区宽度调制效应是指集电结上电压的变化引起有效基区宽度发生变化的效应,也称为“Early”效应。在正向作用区,基区宽度调制对器件特性所起的总效应表现在使饱和电流 I_S、电流增益 β 和正向渡越时间 τ 发生变化。

在 EM3 模型中考虑基区宽度调制效应,要增加一个模型参数,即 Early 电压 U_A。Early 电压的测量方法如图 3.2.8 所示,U_A 由晶体管输出特性曲线和电压轴的截距求得。

图 3.2.8　根据 I_C 随 U_{CE} 变化曲线近似定义 U_A(图中未按比例)

由 U_A 可以求出基区宽度调制效应对基区宽度 W 的影响,即

$$W(U_{BC}) = W(0)\left(1+\frac{U_{BC}}{U_A}\right) \tag{3.2.36}$$

其中 $W(0)$ 表示 $U_{BC}=0$ 时的基区宽度。

基区宽度调制效应对 I_S, β 和 τ 的影响可由下面公式表示,通常 I_S 和 β 与基区宽度成反比,τ 与基区宽度的平方成正比,即有

$$I_S(U_{BC}) = \frac{I_S(0)}{1+\dfrac{U_{BC}}{U_A}} \tag{3.2.37}$$

$$\beta(U_{BC}) = \frac{\beta(0)}{1+\dfrac{U_{BC}}{U_A}} \tag{3.2.38}$$

$$\tau(U_{BC}) = \tau(0)\left(1+\frac{U_{BC}}{U_A}\right)^2 \tag{3.2.39}$$

上述各式中

$$U_A = \left[\frac{1}{W(0)}\frac{dW}{dU_{BC}}\right]^{-1}_{U_{BC}=0} \tag{3.2.40}$$

$I_S(0)$、$\beta(0)$ 和 $\tau(0)$ 均为集电结零偏置时的对应值。

2．β 随电流的变化

我们主要考虑 β 随集电极电流的变化。可把 β 与 I_C 的关系分为三个区:小电流区、中电流区和大电流区,如图 3.2.9 所示。在小电流区,β 随 I_C 增加而增加;在大电流区,β 随 I_C 增加而下降;在中电流区,β 为常量,这时又称 β 为 β_M。

在小电流区,β 的跌落是由于额外的 I_B 分量所引起的。I_B 分量主要是表面载流子复合和发射极-基极空间电荷层内的载流子复合的结果。通常可把 I_B 表示式写为

$$I_B = \frac{I_S}{\beta_{FM}}\left[\exp\left(\frac{qU_{BE}}{kT}\right)-1\right] + \frac{I_S}{\beta_{RM}}\left[\exp\left(\frac{qU_{BC}}{kT}\right)-1\right]$$

图 3.2.9　β 与 I_C 关系

$$+ C_2 I_S \exp\frac{qU_{B'E'}}{n_{EL}kT} - 1 + C_4 I_S \exp\frac{qU_{B'C'}}{n_{CL}kT} - 1 \qquad (3.2.41)$$

该式后两项为额外 I_B 所增加的分量,从而引入了 4 个模型参数: C_2, C_4, n_{EL}(低电流正向区发射系数)和 n_{CL}(低电流反向区发射系数)。在图 3.2.7 的 EM3 模型中,用两个二极管 D_1 和 D_2 模拟这两个额外的 I_B 分量。

在大电流区, β 的下降是由于注入到基区中的少数载流子超过了基区中平衡的多数载流子,而引起注入效率降低的结果。在集电极电流 I_C 较大时, I_C 和 $U_{B'E'}$ 的关系为

$$I_C = \frac{I_S}{1 + \theta\exp\dfrac{qU_{B'E'}}{2kT}} \exp\frac{qU_{B'E'}}{kT} - 1 \qquad (3.2.42)$$

其中, θ为新模型参数,通常约为 10^{-7}至 10^{-6}。

3. 正向渡越时间 τ_F 随电流的变化

晶体管的正向渡越时间 τ_F 是由发射极延迟时间、基区渡越时间 τ_b 和 EB 结、CB 结空间电荷区渡越时间组成。其中基区渡越时间 τ_b 起主要作用。当集电极电流超过一定数值 I_{KF}时,正向渡越时间随集电极电流增加而增加。τ_F 随电流的变化是由于基区宽度的有效增加而引起的。当 $I_C \geq I_{KF}$时,实际正向渡越时间表达式为

$$\tau_F = \tau_{FL}\left[1 + \frac{1}{4}\left(\frac{L_E}{W}\right)^2 \left(\frac{I_C}{I_{KF}} - 1\right)^2\right] \qquad (3.2.43)$$

其中 τ_{FL}为低电流时正向渡越时间值, L_E 为发射区最小宽度, W 为基区宽度,而 I_{KF}为当 τ_F 刚开始增大时的电流。由(3.2.43)式描述的 τ_F 随 I_C 的变化关系,示于图 3.2.10。

4. 模型参数随温度的变化

晶体管中许多参数与温度条件有关,并随温度变化而变化。

在 BJT 模型中,饱和电流 I_S 和温度的关系由下式决定:

$$I_S(T) = I_S(T_0)\left(\frac{T}{T_0}\right)^{XTI}\exp\frac{qE_g(T - T_0)}{kTT_0} \qquad (3.2.44)$$

图 3.2.10　τ_F 随 I_C 变化的曲线

其中 T_0 是常温(300K), E_g 是禁带宽度, XTI 是饱和电流温度指数, E_g 和 XTI 都是模型参数。

正向电流增益(或反向电流增益)的温度关系式为

$$\beta(T) = \beta(T_0)\left(\frac{T}{T_0}\right)^{XTB} \tag{3.2.45}$$

其中 XTB 为模型参数。

正向渡越时间 τ_F 和温度的关系式为

$$\tau_F(T) = \tau_F(T_0)\left(\frac{W(T)}{W(T_0)}\right)^2\left(\frac{T}{T_0}\right)^{1.5} \tag{3.2.46}$$

对某些参数可以采用器件温度系数拟合的随温度变化的经验公式, 即

$$Par(T) = Par(T_0)\left[1 + TC_1(T - T_0) + TC_2(T - T_0)^2\right] \tag{3.2.47}$$

其中 Par 为随温度变化的参数, 可以是 β, $R_{BB'}$, $R_{CC'}$ 等。TC_1 为一阶温度系数, TC_2 为二阶温度系数。

EM3 模型所需的模型参数, 除了前面 EM2 模型的 15 个参数外, 还需增加如下模型参数: Early 电压 U_A, 考虑 β 随 I_C 变化的各模型参数 β_M、C_2、n_{EL}、θ, 考虑 β_R 随 I_C 变化的各模型参数 β_{RM}、C_4、n_{CL}、θ, 考虑 τ 随 I_C 变化的各模型参数 L_E/W、I_{KF} 以及温度系数 TC_1、TC_2 等。如果考虑到更多的二阶效应, 还需增加相应的模型参数。表 3.2.1 中列出了 SPICE 程序中 BJT 晶体管的全部 40 个模型参数。

表 3.2.1 SPICE 中 BJT 管模型参数表

序号	符号	SPICE 关键字	名　　称	隐含值	单位	举例
1	I_S	IS	饱和电流	10^{-16}	A	10^{-15}A
2	β_F	BF	正向电流增益	100		80
3	β_R	BR	反向电流增益	1		1.5
4	n_F	NF	正向电流发射系数	1		1
5	n_R	NR	反向电流发射系数	1		1
6	c_2	ISE(c_2I_S)	B-E 结泄漏饱和电流	0	A	
7	c_4	ISC(c_4I_S)	B-C 结泄漏饱和电流	0	A	
8	I_{KF}	IKF	正向 β大电流下降点	∞	A	0.01A
9	I_{RF}	IRF	反向 β大电流下降点	∞	A	10^{-13}A
10	n_{EL}	NE	B-E 结泄漏发射系数	1.5		2
11	n_{CL}	NC	B-C 结泄漏发射系数	2		2
12	U_A	VAF	正向欧拉电压	∞	V	200V
13	U_B	VAR	反向欧拉电压	∞	V	200V
14	$R_{CC'}$	RC	集电极电阻	0	Ω	10Ω
15	$R_{EE'}$	RE	发射极电阻	0	Ω	1Ω
16	$R_{BB'}$	RB	基极电阻	0	Ω	10Ω
17	R_{BM}	RBM	大电流时最小基极欧姆电阻	RB	Ω	10Ω
18	I_{RB}	I_{RB}	基极电阻下降到最小值的 1/2 时的电流	∞	A	0.1A
19	τ_F	TF	理想正向渡越时间	0	s	1ns

续表

序号	符号	SPICE 关键字	名　　称	隐含值	单位	举例
20	τ_R	TR	理想反向渡越时间	0	s	10ns
21	$X_{\tau F}$	XTF	TF 随偏置变化系数	0		
22	$U_{\tau F}$	VTF	描述 TF 随 U_{BC} 变化的电压	∞		
23	$I_{\tau F}$	ITF	TF 的大电流参数	0	A	
24	$P_{\tau F}$	PTF	$f=\frac{1}{2\pi\tau_F}$ 时超前相位	0		
25	C_{JEo}	CJE	B-E 结零偏置耗尽电容	0	F	2pF
26	ϕ_E	VJE	B-E 结内建电势	0.75	V	0.70V
27	m_E	MJE	B-E 结梯度因子	0.33		0.35
28	C_{JCo}	CJC	B-C 结零偏置耗尽电容	0	F	2pF
29	ϕ_C	VJC	B-C 结内建电势	0.75	V	0.70V
30	m_C	MJC	B-C 结梯度因子	0.33		0.35
31	C_{SUB}	CJS	C-衬底结零偏置电容	0	F	5pF
32	ϕ_S	VJS	衬底结内建电势	0.75	V	0.70V
33	m_S	MJS	衬底结梯度因子	0.33		0.35
34	FC	FC	正偏压耗尽电容公式中的系数	0.5		
35	X_{CJC}	XCJC	B-C 结耗尽电容连到基极内节点的百分数	1		
36	X_{TB}	XTB	BETA 的温度指数	0		
37	X_{TI}	XTI	饱和电流温度指数	3		
38	E_g	EG	IS 温度效应中的禁带宽度	1.11	(硅)eV	
39	K_f	KF	闪烁噪声系数	0		
40	a_f	AF	闪烁噪声指数	1		

3.2.4　GP 模型

Gummel-Poon(GP)模型是 Gummel 和 Poon 于 1970 年提出的,它基本上是与 EM3 模型等价的一种模型。EM3 模型所模拟的二阶效应,如基区宽度调制、τ_F 随电流变化以及随环境温度变化等等是分别进行处理的,模型公式也是一个个地进行修正的。GP 模型所涉及的同样二阶效应却是把它们放在一起来处理,这种统一处理比 EM3 模型能提供一个更精确而且更完整的模型。所以,GP 模型也是一种更加数学化的模型。

美国著名的电路模拟程序 SPICE 所采用的二阶 BJT 模型就是一种改进的 GP 模型。

在用 SPICE 程序做电路分析时,如果模型语句中没有定义 GP(或称 EM3)模型参数,那么程序自动认为模型为 EM2 模型。一个典型的 EM2 型 BJT 管的模型语句如下:

```
.MODEL  MQ  NPN  IS= IE- 15  RB= 20  BF= 80
+ BR= 1  CJC= 2p  CJE= 2p  TF= 5ns
```

模型语句中没有指定的模型参数,程序按其隐含值计算。

3.3 MOS 场效应晶体管模型

MOS 晶体管是金属-氧化物-半导体场效应晶体管，是目前集成电路中最常用的半导体器件。最初集成电路中 MOS 晶体管模型采用一种较为简单的一维近似模型，随着集成电路规模的不断增大和 IC 工艺的迅速发展，MOS 晶体管的沟道长度和宽度不断缩小，晶体管中的二阶效应更为显著，因此常常需用二维甚至三维的数值解法才能满足精度的要求，MOS 晶体管的模型也越来越复杂。例如 SPICE 程序中 MOS 场效应管模型有 6 种，分别是 MOS1～6。其中 MOS1、MOS2 是长沟道模型，适用于 2 μm 以上的 MOS 管。MOS1 是最简单的一维近似模型；MOS2 是一种二维模型，它考虑到沟道对阈值电压的影响，表面电场对载流子迁移率的影响，载流子极限速度引起的饱和以及沟道长度调制等二阶效应。MOS3 模型是一种半经验模型，主要用于小尺寸短沟道的 MOS 器件，它考虑到短沟道效应和静电反馈效应，阈值电压、沟道电流和饱和电压等表达式都与 MOS2 模型有很大不同。MOS4～6 属于更小尺寸的 MOS 模型，MOS4，MOS5 也称为 BSIM1 和 BSIM2 模型，它们都是为适应亚微米器件模拟的需要而建立的，具体特性和公式请参阅有关文献。在 SPICE 程序的例题库中，有这些器件模型的应用实例，提供给使用者参考。这里，我们只着重介绍 MOS1 和 MOS2 模型的基本公式和相应的模型参数。

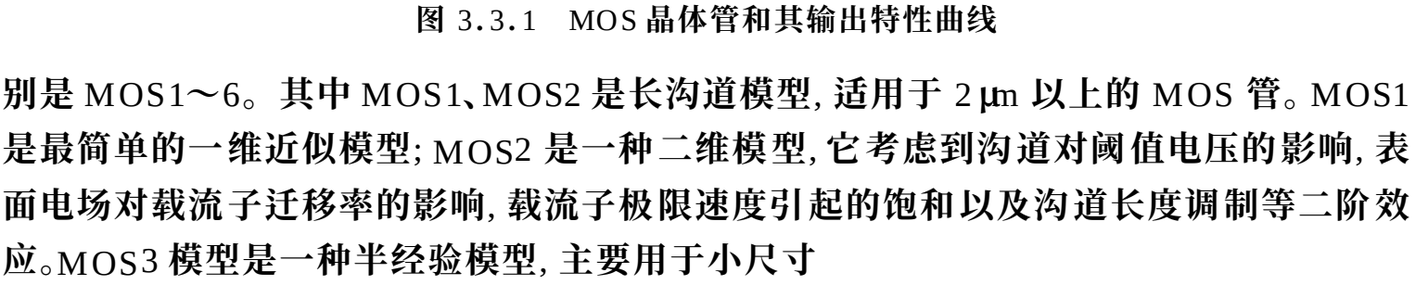


图 3.3.1　MOS 晶体管和其输出特性曲线

集成电路中的 MOS 晶体管是个四端器件，它的四个引出端分别是：栅极(G)、漏极(D)、源极(S)和衬底(B)。图 3.3.1 中示出了 N 沟道 MOS 晶体管以及它的输出特性曲线。N 沟道的 MOS 晶体管的模型如图 3.3.2 所示。

图 3.3.2　MOS 晶体管模型

MOS 晶体管模型中 I_{DS} 为沟道电流，用非线性电流源表示；R_D 和 R_S 分别是漏极和源极的欧姆电阻；5 个电容 C_{GD}，C_{GS}，C_{GB}，C_{BD} 和 C_{BS} 反映 MOS 管的电荷存储效应；漏-衬底和源-衬底 PN 结分别用两个二极管表示，I_{BD} 和 I_{BS} 分别为二极管电流，由下列公式决定：

$$I_{BD} = I_S \exp\left(\frac{qU_{BD}}{kT}\right) - 1 \tag{3.3.1}$$

$$I_{BS} = I_S \exp\left(\frac{qU_{BS}}{kT}\right) - 1 \tag{3.3.2}$$

3.3.1 非线性电流源 I_{DS}

当在 MOS 晶体管上加上不同的栅源电压 U_{GS} 和漏源电压 U_{DS} 时，它可工作在线性区、饱和区和截止区。以下我们将介绍 SPICE 程序中，MOS1 和 MOS2 晶体管模型工作在线性区和饱和区时的沟道电流 I_{DS} 的计算公式。

1. MOS1 模型(Level1)

(1) 线性区：$U_{GS} > V_{TH}$ 且 $U_{DS} < U_{GS} - V_{TH}$

$$I_{DS} = K_p \frac{W}{L - 2L_d}\left[(U_{GS} - V_{TH}) - \frac{U_{DS}}{2}\right]U_{DS}(1 + \lambda U_{DS}) \tag{3.3.3}$$

$$V_{TH} = V_{To} + \gamma\left(\sqrt{2\phi - V_{BS}} - \sqrt{2\phi}\right) \tag{3.3.4}$$

其中，L_d ——横向扩散长度；

V_{To} ——$U_{BS} = 0$ 时的阈值电位；

K_p ——跨导参数；

λ ——沟道长度调制系数；

W ——沟道宽度；

L ——沟道长度；

γ ——体材料的阈值参数；

ϕ ——费米势。

这些参数中，K_p，γ 和 ϕ 可以作为模型参数直接输入，也可以用下面公式计算：

$$K_p = \mu C'_{ox} \tag{3.3.5}$$

$$\gamma = \frac{\sqrt{2\varepsilon q N_{SUB}}}{C'_{ox}} \tag{3.3.6}$$

$$2\phi = 2\frac{kT}{q}\ln\frac{N_{SUB}}{n_i} \tag{3.3.7}$$

式中，μ ——沟道中载流子迁移率；

C'_{ox} ——单位面积栅氧化层电容；

N_{SUB} ——衬底杂质浓度；

ε ——硅的介电常数；

n_i ——硅的本征载流子浓度。

(2) 饱和区：$U_{GS} > V_{TH}$ 且 $U_{DS} > U_{GS} - V_{TH}$

$$I_{DS} = \frac{K_p}{2}\frac{W}{L - 2L_d}(U_{GS} - V_{TH})^2(1 + \lambda U_{DS}) \tag{3.3.8}$$

在饱和区,漏区与源区间的沟道产生夹断现象,使电流与电压关系呈饱和状态。

2. MOS2 模型(Level2)

(1) 线性区:

$$I_{DS} = \frac{K_p}{1 - \lambda U_{DS}}\frac{W}{L - 2L_d}\left[U_{GS} - U_{FB} - 2\phi - \frac{U_{DS}}{2}\right]U_{DS} - \frac{2}{3}\gamma\left[(U_{DS} - U_{BS} + 2\phi)^{1.5} - (-U_{BS} + 2\phi)^{1.5}\right] \tag{3.3.9}$$

其中 U_{FB} 为平带(flat-band)电压。

(2) 饱和区:

$$I_{DS} = I_{D,sat}\frac{1}{1 - \lambda U_{DS}} \tag{3.3.10}$$

式中 $I_{D,sat}$ 是(3.3.9)式在 $U_{DS} = U_{D,sat}$ 时的值,而 $U_{D,sat}$ 是发生夹断时的漏电压,也称为漏夹断电压或饱和电压,其值为

$$U_{D,sat} = U_{GS} - U_{FB} - 2\phi + \frac{1}{2}\left[1 - \sqrt{1 - \frac{2}{\frac{1}{2}}(U_{GS} - U_{FB})}\right] \tag{3.3.11}$$

3.3.2 电荷存储效应

考虑 MOS 晶体管的电荷存储效应,我们主要讨论覆盖电容、结电容和侧壁电容等产生的 MOS 管极间的等效电容 C_{GS}, C_{GD}, C_{SB} 和 C_{DB},它们对 MOS 晶体管的频域和时域特性有较大的影响。

1. 栅源和栅漏的覆盖电容

一般栅极与源区和漏区总有一个覆盖区,因而产生一定的覆盖电容,栅源和栅漏覆盖电容分别为

$$C_{GSo} = WX_S C'_{ox} \tag{3.3.12}$$

$$C_{GDo} = WX_D C'_{ox} \tag{3.3.13}$$

其中,X_S 和 X_D 分别为栅源和栅漏的覆盖长度。

2. 结电容

衬底-沟道结电容 C_{JBC}、衬底-源结电容 C_{JSB} 和衬底-漏结电容 C_{JDB} 均是结区空间电荷耗尽层电容,它们与二极管中的 PN 结耗尽层电容类似,其表示式为

$$C_{JBC} = WL\frac{C_J}{\left(1 - \frac{U_{BC}}{\phi}\right)^{m_J}} \tag{3.3.14}$$

$$C_{JSB} = A_S\frac{C_J}{\left(1 - \frac{U_{BS}}{\phi}\right)^{m_J}} + P_S\frac{C_{JSW}}{\left(1 - \frac{U_{BS}}{\phi}\right)^{m_{JSW}}} \tag{3.3.15}$$

$$C_{JDB} = A_D\frac{C_J}{\left(1 - \frac{U_{BD}}{\phi}\right)^{m_J}} + P_D\frac{C_{JSW}}{\left(1 - \frac{U_{BD}}{\phi}\right)^{m_{JSW}}} \tag{3.3.16}$$

其中, C_J —— 每平方米的零偏置衬底电容;

ϕ—— 衬底结内建电势;

m_J —— 衬底结电容梯度因子;

C_{JSW} —— 每单位周边长度的零偏置衬底电容;

m_{JSW} —— C_{JSW} 的梯度因子;

A_D, P_D —— 漏区面积和周长;

A_S, P_S —— 源区面积和周长。

3. 极间等效电容

MOS 晶体管源、漏、栅和衬底间的极间电容是由覆盖电容、结电容和周边电容等并联在一起形成的等效电容, 其表示式如下。

线性区:

$$C_{GS} = C_{GSo} + \frac{1}{2}C_{ox} \tag{3.3.17}$$

$$C_{GD} = C_{GDo} + \frac{1}{2}C_{ox} \tag{3.3.18}$$

$$C_{SB} = C_{JSB} + \frac{1}{2}C_{JBC} \tag{3.3.19}$$

$$C_{DB} = C_{JDB} + \frac{1}{2}C_{JBC} \tag{3.3.20}$$

其中

$$C_{ox} = WLC'_{ox} \tag{3.3.21}$$

饱和区:

$$C_{GS} = C_{GSo} + \frac{2}{3}C_{ox} \tag{3.3.22}$$

$$C_{GD} = C_{GDo} \tag{3.3.23}$$

$$C_{SB} = C_{JSB} + \frac{2}{3}C_{JBC} \tag{3.3.24}$$

$$C_{DB} = C_{JDB} \tag{3.3.25}$$

以上所述的 MOS1 和 MOS2 晶体管模型是适用于一般长沟道 MOS 管的一阶近似模型, 其模型参数有近 40 个, 列于表 3.3.1 中。

在用 SPICE 程序做模拟时, 一个典型的 MOS1 管和它的模型语句是:

```
M20  6  7  4  0  MOD  W= 250u  L= 5u
.MODEL  MOD  NMOS  (VTo= 0.5  PHI= 0.7  Kp= 1.0E- 6
+ GAMMA= 1.83  LAMBDA= 0.115  LEVEL= 1  CGSo= 1n
+ CGDo= 1n  CBD= 50p  CBS= 50p)
```

其中, M20 是 MOS 管名称; 6, 7, 4, 0 分别表示 MOS 管的漏、栅、源和衬底的节点号; MOD 是模型名称。

表 3.3.1　SPICE 中 MOS 管模型参数表

序号	符号	SPICE 关键字	名　　称	隐含值	单位	举例
1	V_{To}	VTo	零偏阈值电位	1.0	V	1.0V
2	K_p	Kp	跨导参数	2×10^{-5}	A/V^2	$3\times10^{-5}A/V^2$
3	γ	GAMMA	体材料阈值参数	0.0	$V^{\frac{1}{2}}$	$0.35V^{\frac{1}{2}}$
4	2ϕ	PHI	表面电势	0.6	V	0.65V
5	λ	LAMBDA	沟道长度调制系数	0.0	V^{-1}	$0.02V^{-1}$
6	t_{ox}	TOX	氧化层厚度	1×10^{-7}	m	$1\times10^{-7}m$
7	N_b	NSUB	衬底掺杂浓度	0.0	cm^{-3}	$1\times10^{15}cm^{-3}$
8	N_{SS}	NSS	表面态密度	0.0	cm^{-2}	$1\times10^{10}cm^{-2}$
9	N_{FS}	NFS	快表面态密度	0.0	cm^{-2}	$1\times10^{10}cm^{-2}$
10	N_{eff}	NEFF	总沟道电荷系数	1		5
11	X_J	XJ	结深	0.0	m	$1\times10^{-6}m$
12	L_d	LD	横向扩散长度	0.0	m	$0.8\times10^{-6}m$
13	T_{PG}	TPG	硅材料类型	1(硅栅)		0(铝栅)
14	μ_b	Uo	载流子表面迁移率	600	$cm^2/(V\cdot s)$	$700cm^2/(V\cdot s)$
15	U_c	UCRIT	迁移率下降时临界电场	1×10^4	V/cm	$1\times10^4V/cm$
16	U_e	UEXP	迁移率下降时临界电场指数	0.0		0.1
17	U_t	UTRA	迁移率下降时横向电场系数	0.0		0.5
18	U_{max}	UMAX	载流子最大漂移速度	0.0	m/s	$5\times10^4m/s$
19	δ	DELTA	阈值电压的沟道宽度效应系数	0.1		1.0
20	X_{QC}	XQC	漏端沟道电荷分配系数	0.0		0.4
21	η	ETA	静态反馈系数	0.0		1.0
22	θ	THETA	迁移率调制系数	0.0	V^{-1}	$0.05V^{-1}$
23	A_F	AF	闪烁噪声指数	1.0		1.2
24	K_F	KF	闪烁噪声系数	0.0		1×10^{-26}
25	I_S	IS	衬底结饱和电流	1×10^{-14}	A	$1\times10^{-15}A$
26	J_S	JS	I_S/每平方米	0.0	A	$1\times10^{-8}A$
27	ϕ	PB	衬底结电势	0.80	V	0.75V
28	C_J	CJ	零偏衬底电容/每平方米	0.0	F/m^2	$2\times10^{-4}F/m^2$
29	m_J	MJ	衬底结电容梯度因子	0.5		0.5
30	C_{JSW}	CJSW	零偏衬底电容/单位周边长度	0.0	F/m	$1\times10^{-9}F/m$
31	M_{JSW}	MJSW	衬底周边电容梯度因子	0.33		0.33

续表

序号	符号	SPICE 关键字	名　　称	隐含值	单位	举例
32	FC	FC	正偏耗尽电容系数	0.5		0.5
33	C_{GBo}	CGBo	G-B 间覆盖电容/单位沟道宽度	0.0	F/m	2×10^{-10}F/m
34	C_{GDo}	CGDo	G-D 间覆盖电容/单位沟道宽度	0.0	F/m	4×10^{-11}F/m
35	C_{GSo}	CGSo	G-S 间覆盖电容/单位沟道宽度	0.0	F/m	4×10^{-11}F/m
36	R_D	RD	漏极欧姆电阻	0.0	Ω	10.0Ω
37	R_S	RS	源极欧姆电阻	0.0	Ω	10.0Ω
38	R_{sh}	RSH	漏源扩散区薄层电阻	0.0	Ω	30.0Ω
39	C_{BD}	CBD	零偏 B-D 结电容			
40	C_{BS}	CBS	零偏 B-S 结电容			

3.4　结型场效应晶体管模型

结型场效应晶体管(简称 JFET)和 MOS 晶体管类似,都是由栅极电位控制沟道区中载流子从而改变源漏电流的器件。所不同的是 JFET 是靠反向偏置 PN 结的耗尽层使导电沟道夹断的。结型场效应晶体管有三个电极:源极(S)、漏极(D)和栅极(G),也有与 MOS 晶体管类似的特性曲线。图 3.4.1 中表示出了 N 沟道结型场效应晶体管的模型。图中 I_{DS} 为非线性电流源;R_D, R_S 分别是漏极和源极的串联欧姆电阻;电容 C_{GD} 和 C_{GS} 反映两个栅结的电荷存储效应;栅源和栅漏 PN 结分别用两个二极管 D_D 和 D_S 表示,二极管电流公式为

$$I_{GD}=\begin{cases}-I_S+U_{GD}G_{min} & U_{GD}\leq-5\dfrac{kT}{q}\\ I_S\exp\dfrac{qU_{GD}}{kT}-1+U_{GD}G_{min} & U_{GD}>-5\dfrac{kT}{q}\end{cases}\qquad(3.4.1)$$

图 3.4.1　结型场效应晶体管模型

$$I_{GS}=\begin{cases}I_S\exp\dfrac{qU_{GS}}{kT}-1+U_{GS}G_{min} & U_{GS}>-5\dfrac{kT}{q}\\ -I_S+U_{GS}G_{min} & U_{GS}\leq-5\dfrac{kT}{q}\end{cases}\qquad(3.4.2)$$

式中各参数的意义与前面介绍的 SPICE 模型参数相同。G_{min} 仍是为帮助收敛而并在 PN 结两端的小电导,隐含值是 10^{-12}S。

对于 N 沟道 JFET 管,非线性沟道电流 I_{DS} 在正向工作区的计算公式是

$$I_{DS}=\begin{cases}0 & U_{GS}-V_{To}\le 0\\ \beta(U_{GS}-V_{To})^2(1+\lambda U_{DS}) & 0<U_{GS}-V_{To}\le U_{DS}\\ \beta U_{DS}\,2(U_{GS}-V_{To})-U_{DS}\,(1+\lambda U_{DS}) & 0<U_{DS}<U_{GS}-V_{To}\end{cases}\qquad(3.4.3)$$

反向区计算公式是

$$I_{DS}=\begin{cases}0 & U_{GD}-V_{To}\le 0\\ -\beta(U_{GD}-V_{To})^2(1-\lambda U_{DS}) & 0<U_{GD}-V_{To}\le -U_{DS}\\ \beta U_{DS}\,2(U_{GD}-V_{To})+U_{DS}\,(1-\lambda U_{DS}) & 0<-U_{DS}<U_{GD}-V_{To}\end{cases}\qquad(3.4.4)$$

其中，β——跨导参数；

V_{To}——阈值电位；

λ——沟道长度调制系数。

由于 JFET 在正常工作时，两个二极管 D_D 和 D_S 均处于反向偏置，故考虑 JFET 的电荷存储效应时只讨论势垒电容上的电荷存储。势垒电容的计算公式如下：

$$C_{GS}=\begin{cases}C_{GSo}\left(1-\dfrac{U_{GS}}{\phi}\right)^{-\frac{1}{2}} & U_{GS}<FC\times\phi\\ \dfrac{C_{GSo}}{F_2}\left(F_3+\dfrac{U_{GS}}{2\phi}\right) & U_{GS}\ge FC\times\phi\end{cases}\qquad(3.4.5)$$

$$C_{GD}=\begin{cases}C_{GDo}\left(1-\dfrac{U_{GD}}{\phi}\right)^{-\frac{1}{2}} & U_{GD}<FC\times\phi\\ \dfrac{C_{GDo}}{F_2}\left(F_3+\dfrac{U_{GD}}{2\phi}\right) & U_{GD}\ge FC\times\phi\end{cases}\qquad(3.4.6)$$

其中，

$$F_2=(1-FC)^{1+m}\qquad F_3=1-FC(1+m)\qquad(3.4.7)$$

C_{GSo}——零偏置 G-S 结电容；

C_{GDo}——零偏置 G-D 结电容；

φ——栅结内建电势；

m——电容梯度因子；

FC——正偏耗尽电容公式中系数。

结型场效应管的模型参数有 13 个，列于表 3.4.1 中。

表 3.4.1　SPICE 中 JFET 管模型参数表

序号	符号	SPICE 关键字	名称	隐含值	单位	举例
1	V_{To}	VTo	阈值电压	-2	V	
2	β	BETA	跨导参数	10^{-4}	A/V^2	
3	λ	LAMBDA	沟道长度调制系数	0	V^{-1}	
4	R_D	RD	漏极欧姆电阻	0	Ω	10Ω
5	R_S	RS	源极欧姆电阻	0	Ω	10Ω
6	C_{GSo}	CGS	零偏 G-S 结电容	0	F	
7	C_{GDo}	CGD	零偏 G-D 结电容	0	F	

续表

序号	符号	SPICE 关键字	名称	隐含值	单位	举例
8	ϕ	PB	栅结内建电势	1	V	
9	m	M	电容梯度因子	0.33		0.35
10	I_S	IS	栅 PN 结饱和电容	10^{-14}	A	
11	FC	FC	正偏耗尽电容系数	0.5		
12	K_f	KF	闪烁噪声系数	0		
13	a_f	AF	闪烁噪声指数	0		

3.5 宏 模 型

所谓宏模型是电子电路或系统中的一个子网络或子系统的简化等效表示。它可以是一个等效电路,也可以是一组数学方程、一组多维数表,或者是表达更复杂电路的某种符号形式。宏模型的特点是,在一定精度范围内,其端口特性和原子网格或子系统的端口特性相同或近似相同,而其结构复杂程度明显下降,所含元件数和节点数也大大减少。例如,一个运算放大器由 20 个晶体管组成,每个晶体管模型由 13 个元件构成,则总数为 260 个元件。而它的一种宏模型仅用 11 个元件就可构成,节点和支路数也显著减少。这显然大大降低了对计算机内存的要求,并节省了计算时间。用 SPICE 程序对电路宏模型和元件级模型进行比较,大量计算统计表明,前者所用 CPU 时间比后者节省 6～10 倍以上,节点数和支路数大约减少 5 倍以上。因此,宏模型在大型电路,特别是在 LSI 和 VLSI 的计算机辅助分析与设计中具有十分重要的实际意义。

建立宏模型的基本要求是:

1) 按照精度要求,准确地模拟原电路的电特性。

2) 宏模型本身的电路结构要尽可能简单。

3) 建立宏模型的过程要尽可能简化。

常用的建立宏模型的方法主要是两种。一种是简化电路法(simplification),另一种是端口特性构造法(build-up)。下面简要介绍这两种方法。

3.5.1 简化电路法

简化电路法是将原电路中对整个电路性能影响不大的元件去掉,使原电路得到简化,这个简化后的电路称为原电路的简化宏模型。

建立简化宏模型是用求灵敏度的方法。首先计算电路中各元件的灵敏度,然后比较这些元件的灵敏度,当元件 P_1 的灵敏度高于元件 P_2 的灵敏度 10 倍以上,就认为 P_2 可以去掉。这样连续删除那些灵敏度较低、影响不大的元件,保留灵敏度较高、起主导作用的元件,以使原电路得到简化。例如,用这种方法求图 3.5.1(a)所示电路的小信号频域宏模型。这个电路有 12 个元器件,节点数为 7。简化后的宏模型如图 3.5.1(b)所示。它仅含有 5 个元器件,节点数为 3。

用简化电路法构造宏模型时,由于简化过程主要是计算元件灵敏度和比较灵敏度,所

图 3.5.1　高频放大器及其宏模型

以很容易用计算机实现,方法简便,易于自动化,这是它的优点。它的缺点是,由于宏模型中的元件就是原电路中的元件,因此简化的程度受到限制,模型中保留的元件数与精度要求有关,在精度较高时,所含元件数不能显著减少。

3.5.2　端口特性构造法

端口特性构造法是构造一个另外的电路,使其端口特性与原电路的端口特性一致,新电路要比原电路有较大程度的简化,该新电路称为原电路的端口特性宏模型。构造端口特性宏模型时,应首先分析原电路的工作原理和内部电路各部分的功能,按照功能将电路划分成若干子电路块,用尽可能简单的电路形式和元件去模拟和构造各子电路块的功能,最后将各部分子电路块有机地组合起来构成整个电路的宏模型。一般,在端口部分可以采用有源器件来构造,以便更精确地模拟端口特性;构造其它子电路块时应尽量少采用有源器件,从而降低宏模型的复杂程度。宏模型构造的精度和简化度在很大程度上取决于设计人员的电路理论水平。目前国内外都有不少电路工作者在从事集成电路宏模型的工作,建立宏模型库和宏模型的自动生成工具,以适应当今大规模集成电路的模拟需要。

目前,模拟集成电路的宏模型多由构造法建立,例如运算放大器、模拟相乘器、比较器、模拟开关、开关电源电路等等的宏模型都是采用构造法实现的,并已商用化了。下面我们以运算放大器为例,对构造法建模做一些具体说明。

描述一个运算放大器的基本参数是:输入偏置电流 I_B,输入失调电流 I_{IO},输入失调电压 U_{IO},输入阻抗 Z_I,输出阻抗 Z_O,差模电压增益 $A_{VD}(j\omega)$,共模抑制比 $K_{CMR}(j\omega)$,摆率等等。所建立的宏模型应能模拟这些基本性能参数。用构造法建模可以按直流特性、小信号交流特性和大信号特性分别建立运算放大器的宏模型,也可以建立同时实现上述全部特性的整体宏模型。我们在此只介绍按特性分别建模的方法。

1. 运放直流特性宏模型

运算放大器的直流特性宏模型主要模拟运算放大器的输入偏置电流、输入失调电流和电压,差模和共模输入电阻,直流增益,共模抑制比及输出电阻等。图 3.5.2 是一种运放的直流特性宏模型。在图 3.5.2 所示的直流特性宏模型中,各元件的作用如下:

图 3.5.2　运算放大器直流特性宏模型

(1) 电流源 I^+ 和 I^-，分别模拟两个输入端所需的输入电流。根据运放技术手册中给出的 I_B 和 I_{IO} 参数，宏模型中的 I^+ 和 I^- 的值可由下式决定：

$$\begin{aligned} I^+ \text{（或 } I^- \text{）} &= I_B \\ I^- \text{（或 } I^+ \text{）} &= I_B + I_{IO} \end{aligned} \tag{3.5.1}$$

(2) 电压源 E_{IO}，模拟手册中的输入失调电压 U_{IO}，它是指为使运放的输出电压为零必须在输入端加入的差动直流电压。即

$$E_{IO} = U_{IO} \tag{3.5.2}$$

(3) 电阻 R_{ID}，模拟差模输入电阻。由于电阻 R_C 的取值远小于 R_{ID}，故宏模型两输入端间所呈现的电阻近似为 R_{ID}，令该电阻值等于手册中给定的差模输入电阻即可。

(4) 电阻 R_{IC}^+ 和 R_{IC}^-，分别模拟同相和反相输入端的共模输入电阻，通常这两个电阻值相同，并令其等于手册中相应数值。

(5) 开环差模增益 A_{VD}，宏模型中，VCCS 的 $g_{m1}U_{ID}$ 反映输入电压和输出电压的关系。输出电压 $U_O = g_{m1}U_{ID}R_O$。如果不考虑 R_C 和 E_{IO} 的影响，则 U_{ID} 就是输入差模电压 U_I，那么宏模型电路的差模增益就是 $g_{m1}R_O$。

(6) 共模抑制比 K_{CMR}，定义为开环差模电压增益 A_{VD} 与开环共模电压增益 A_{VC} 的比，即

$$K_{CMR} = \frac{A_{VD}}{A_{VC}} \tag{3.5.3}$$

在宏模型中，等效差模输入电压 $\Delta U_I = \dfrac{\Delta U_O}{A_{VD}}$，由(3.5.3)式，得

$$\Delta U_{ID} = \frac{A_{VC}\,\Delta U_{IC}}{A_{VD}} = \frac{\Delta U_{IC}}{K_{CMR}} \tag{3.5.4}$$

在图 3.5.2 宏模型中，由 VCCS 的 $g_{m2}U_{IC}$ 在 R_C 两端产生的电压来等效 ΔU_{IC} 值，U_{IC} 是共模输入电压，于是

$$g_{m2}U_{IC}R_C = \frac{U_{IC}}{K_{CMR}}$$

或

$$g_{m2} = \frac{1}{K_{CMR}R_C} \tag{3.5.5}$$

由此可以看出，用图 3.5.2 所示直流特性宏模型可以模拟运算放大器的直流特性，宏

模型中各元件参数可直接从运放技术手册中得到。表 3.5.1 中列出运算放大器 μA741 的手册参数和直流宏模型参数。

表 3.5.1　运放 μA741 的手册参数和直流宏模型参数

参数名称	手册参数	宏模型参数
输入偏置电流	$I_B = 80\times 10^{-9}A$	$I^+ = I_B = 80\times 10^{-9}A$
输入失调电流	$I_{ID} = 10\times 10^{-9}A$	$I^- = I_B + I_{ID} = 90\times 10^{-9}A$
差模输入电阻	$R_{ID} = 2M\Omega$	$R_{ID} = 2M\Omega$
共模输入电阻	$R_{IC} = 2\times 10^9\Omega$	$R_{IC}^+ = R_{IC}^- = R_{IC} = 2\times 10^9\Omega$
输出电阻	$R_O = 75\Omega$	$R_O = 75\Omega$
输入失调电压	$U_{IO} = 2mV$	$E_{IO} = U_{IO} = 2mV$
差模开环增益	$A_{VD} = 2\times 10^5$	$g_{m2} = \dfrac{A_{VD}}{R_O} = 2.76\times 10^3 S$
共模抑制比	$K_{CMR} = 31623$	$g_{m1} = \dfrac{1}{K_{CMR}R_C} = 0.316\times 10^{-6}S$ $(R_C = 100\Omega)$

2．运放交流小信号特性宏模型

运算放大器的交流小信号特性宏模型主要模拟如下交流小信号特性：差模和共模输入阻抗、输出阻抗、频率特性、差模开环电压增益和共模抑制比等。宏模型是由输入级、第一中间级、第二中间级和输出级组成，各级所模拟的运算放大器特性如下。

输入级：模拟差模和共模输入阻抗、共模抑制比；

第一中间级：模拟差模开环电压增益，低频主极点；

第二中间级：模拟高频极点；

输出级：模拟输出阻抗。

宏模型的具体电路如图 3.5.3 所示。其中各元件参数值可由给定的运算放大器的技术数据确定。

图 3.5.3　运放的交流小信号特性宏模型

(1) 输入级

模拟差模输入阻抗的电阻 R_{ID}和电容 C_{ID}的参数值分别取手册中给出的数据即可。

模拟共模输入阻抗的电阻 R_{IC}取自手册数据,电容 C_{IC}利用下式计算:

$$C_{IC} = \frac{1}{2\pi R_{IC} f_{CMR}} \tag{3.5.6}$$

其中,f_{CMR}为共模抑制比下降 3dB 的频率,取手册中数据。

模型输入级中 VCCS 的跨导 g_{m1}的推导与直流特性宏模型 g_{m2}的推导相同,即

$$g_{m1}(f) = \frac{1}{K_{CMR}(f) R_C} \tag{3.5.7}$$

但在交流情况下,共模抑制比 K_{CMR}是频率的函数,所以 VCCS 的跨导 g_{m1}也必然是频率的函数。

(2) 第一中间级

这一级宏模型由受控电流源 $g_{m2}U_I$ 和 R_1 并联电路构成,主要模拟运放小信号交流特性中的差模开环增益 A_{VD}和频率特性中的主极点 ω。

该级的低频增益为 $g_{m2}R_1$,若用这一级模拟整个运算放大器的增益,则有

$$A_{VD} = g_{m2} R_1 \tag{3.5.8}$$

若手册中给出运放的主极点为 ω,则 R_1,C_1 与 ω 有如下关系:

$$R_1 C_1 = \frac{1}{\omega} \tag{3.5.9}$$

(3) 第二中间级

第二中间级由 VCCS、电阻 R_2 和 R_3 以及电容 C_2 组成,模拟高频极点 ω。由于第一中间级已模拟了整个运放的增益,所以这一级的增益应为 1,即

$$g_{m3} R_3 = 1 \tag{3.5.10}$$

高频极点 ω 关系式为

$$R_2 C_2 = \frac{1}{\omega} \tag{3.5.11}$$

可任选电阻 R_3 值,由(3.5.10)式确定 g_{m3};然后按 $R_3 \gg R_2$ 选定 R_2,用(3.5.11)式确定 C_2。

(4) 输出级

这一级模拟运放的输出阻抗,且本级增益为 1,于是有

$$g_{m4} R_O = 1 \tag{3.5.12}$$

运算放大器的输出阻抗 R_O 可从手册中查出。

表 3.5.2 运放 μA741 手册参数和交流小信号宏模型参数

参数名称	手册参数	宏模型参数
差模输入电阻	$R_{ID}=2\times10^{6}\Omega$	$R_{ID}=2\times10^{6}\Omega$
差模输入电容	$C_{ID}=1.4\times10^{-12}F$	$C_{ID}=1.4\times10^{-12}F$
共模输入电阻	$R_{CM}=2\times10^{9}\Omega$	$R_{IC}=2\times10^{9}\Omega$ $C_{IC}=0.256\times10^{-12}F$

续表

参数名称	手册参数	宏模型参数
共模抑制比	$K_{CMR}=3.16\times10^{4}$	$R_C=10\Omega$ $g_m(f)=(300+jf)\times0.1053\times10^{-7}S$
差模开环增益	$A_{VD}=2\times10^{5}$	$R_1=85.2\times10^{3}\Omega$ $g_{m2}=2.35S$
主极点	$\omega=2\pi\times7rad/s$	$R_1=85.2\times10^{3}\Omega$ $C_1=0.267\times10^{-6}F$
高频极点	$\omega=2\pi\times2\times10^{6}rad/s$	$R_2=1\times10^{3}\Omega$ $C_2=79.6\times10^{-12}F$ $R_3=10\Omega$ $g_{m3}=0.1S$
输出电阻	$R_O=75\Omega$	$R_O=75\Omega$ $g_{m4}=0.013S$

综上所述,图 3.5.3 所示交流小信号宏模型可以模拟运放的小信号交流特性。表 3.5.2 中列出了运算放大器 μA741 的交流小信号参数与宏模型参数。

3.5.3 混合构造法

这里介绍一种将简化电路法和构造法相结合构成的运算放大器宏模型。它由输入级、中间级和输出级组成。其中输入级由简化法构成,中间级和输出级由构造法构成。由于输入级是用差分对实现的,能较精确地模拟运放的输入偏置、失调特性、差模和共模输入特性、摆率二阶效应和高频极点。中间级模拟差模电压增益、共模抑制比、低频极点和摆率。输出级模拟了最大电流、最大输出电压和输出阻抗。此宏模型能适用于直流、交流小信号和瞬态分析。其宏模型结构如图 3.5.4 所示。

图 3.5.4 运放宏模型

输入级的晶体管 T_1, T_2 组成差动放大器, T_1 和 T_2 由 EM2 模型来描述。选用不同的基极电流,可以模拟运放的输入偏置电流 I_B、失调电流 I_{IO} 和失调电压 U_{IO}。另外, C_E 模拟摆率二阶效应, C_1 模拟高频极点。

中间级中,受控源 $g_{m1}U_{C1}$ 模拟差模增益,受控源 $g_{m2}V_E$ 模拟共模抑制比,电阻 R_2 模拟直流增益,而 C_2 为补偿电容。

输出级中,输出阻抗由 $R_{O1}+R_{O2}$ 来模拟,二极管 D_1、D_2 及电压控制电压源 E_OU_O 模拟最大输出电流,嵌位电路 D_3、E_C 和 D_4、E_D 模拟最大输出电压。

下面是运算放大器 μA741 采用这种宏模型的模型参数:

$R_{C1}=R_{C2}=5.3k\Omega$	$g_{m2}=5.96\times10^{-9}S$
$C_1=5.46pF$	$g_{m3}=2.36\times10^{2}S$
$R_{E1}=R_{E2}=2.15k\Omega$	$R_2=0.1M\Omega$
$I_E=1.67\times10^{-5}A$	$C_2=0.3pF$
$C_E=3pF$	$R_{O1}=30\Omega$
$R_E=12M\Omega$	$R_{O2}=45\Omega$
$g_{m1}=1.88\times10^{-4}S$	$E_O=1.0V$

3.5.4 运算放大器宏模型的应用实例

一个五阶巴特沃兹有源滤波器,如图 3.5.5 所示。其中的运放 μA741 采用图 3.5.3 所示交流小信号宏模型。对该电路进行频域分析,分析结果示于图 3.5.6 中,图中同时还画出了实测结果。

图 3.5.5 五阶巴特沃兹有源滤波器电路

可以看出,用宏模型代替有源滤波器中的运算放大器,所得到的分析结果与实测结果符合得很好。而含宏模型的电路节点数是实际电路节点数的 1/5,从而大大节省了计算机存储量,加快了计算速度。

目前,许多 EDA 公司(如 Mentor Graphics, Cadence)的仿真系统中都备有宏模型库。以 MicroSim 公司的 PSpice 程序为例,它备有相当丰富的模拟电路宏模型库和数字电路宏模型库。模拟电路宏模型库中包括:通用运算放大器,电压比较器,稳压电源,晶体振荡器,压控电容和电感,磁芯及光耦合器件等等。并为国际上几家著名 IC 公司,如美国 TI 公司(Texas Instruments Incorp.),Analog Devices 公司等建立了专门的模拟电路宏模型库。

图 3.5.6 图 3.5.5 电路的幅频特性

图 3.5.7　OP-07 运放宏模型结构图

数字电路宏模型库中包括: 74 系列数字电路模型, 10K～100K ELD 器件模型, GAL 器件模型, A/D 和 D/A 接口器件模型和 PAL 器件模型等等。

图 3.5.7 中给出了 PSpice 程序中 OP-07 运放大信号宏模型的原理图, 该模型可运用于频域和时域的大信号分析。用 OP-07 构成的一个带通滤波器如图 3.5.8 所示, 用 PSpice 对该带通滤波器进行频域分析, 其频响特性如图 3.5.9 所示, $f_0 \approx 150\text{kHz}$。

图 3.5.8　带通滤波器

3.6　分段线性模型

在电路的计算机辅助分析与设计中, 非线性电路的分析与计算存在一些相当困难的问题。比如, 在不少情况下, 描述非线性特性的函数形式是未知的, 只能给出一组测试数据; 有时, 虽然函数形式是已知的, 但较为复杂, 或者难于在计算机上求数值解, 或者要花费较多的内存和计算时间。如果用一组折线来逼近这种非线性特性, 则可以使非线性问题得以实现和简化, 这组直线段描述的模型称为分段线性模型, 如图 3.6.1 所示。

图中相邻直线段的接点称为连接点, 它是实际非线性特性的实测点或采样点。这些连接点的位置和数目直接关系到分段线性曲线与实际非线性曲线之间误差的大小。每一段直线段可以用线性方程来描述, 也可以用简单的电路元件来实现。也就是说, 分段线性模

图 3.5.9　带通滤波器频响特性

图 3.6.1　分段线性化曲线

图 3.6.2　非线性函数的分段线性逼近

型可以是电路模型,或者是函数表格模型。

1. 分段线性函数表格模型

图3.6.2中表示了一个非线性函数$f(x)$和它的分段线性逼近。设分段线性函数是$P(x)$,在$x\in[x_k, x_{k+1}]$区间内,$P(x)$可以表示为

$$P_k(x) = a_k x + b_k \tag{3.6.1}$$

其中

$$a_k = f(x_k, x_{k+1}) = \frac{f(x_{k+1}) - f(x_k)}{x_{k+1} - x_k} \tag{3.6.2}$$

$$b_k = f(x_k) - x_k a_k \tag{3.6.3}$$

当$k=0,1,2,\cdots,p$时,(3.6.1)式就是分段线性函数的表达式,p是分段数。只要给定分段线段的连接点坐标$(x_k, f(x_k))$,$k=0,1,\cdots,p$,并代入(3.6.1),即可求出分段线性方程组的解。

MicroSim 公司的 PSpice 程序中有一种函数表格模型就属于分段线性模型,它可以是受控电压源或电流源,控制变量可以是电压、电流、频率或拉普拉斯变换等等,并以数据表格的形式描述。

例如图 3.6.3 所示的一个隧道二极管的伏安特性曲线,在 PSpice 中用函数表格模型的描述形式如下:

```
GTUNNEL  5     0     TABLE  {V(5)}  =     (
+                (0,    0)                    (.02,    2.690E-03)
+                (.04,    4.102E-03)          (.06,    4.621E-03)
+                (.08,    4.460E-03)          (.10,    3.860E-03)
+                (.12,    3.079E-03)          (.14,    2.327E-03)
+                (.16,    1.726E-03)          (.18,    1.308E-03)
+                (.20,    1.042E-03)          (.22,    8.734E-04)
+                (.24,    7.544E-04)          (.26,    6.566E-04)
+                (.28,    5.718E-04)          (.30,    5.013E-04)
+                (.32,    4.464E-04)          (.34,    4.053E-04)
+                (.36,    3.781E-04)          (.38,    3.744E-04)
+                (.40,    4.127E-04)          (.42,    5.053E-04)
+                (.44,    6.380E-04)          (.46,    7.935E-04)
+                (.48,    1.139E-03)          (.50,    2.605E-03)
+                (.52,    8.259E-03)          (.54,    2.609E-02)
+                (.56,    7.418E-02)          (.58,    1.895E-01)
+                (.60,    4.426E-01)          )
```

其中,Gtunnel 是受控电流源名称(G 表示电流源,E 表示电压源);5 是该电流源正节点号,0 是负节点号(接地);Table后面大括号内是控制变量的值,它可以是一个表达式,这里的 V(5)表示控制信号是节点 5 的电位;等号后面是由数据对构成的表格数据,每一对数据中,第一个是输入数据,即 V(5)的值,第二个数据是输出数据,即 I(5,0)的值(流经隧道管的电流值)。也就是说,表格中的数据就是分段线性连接点的坐标值。数据对之间的值由(3.6.1)式的线性插值函数来确定。

图 3.6.3　隧道管特性曲线

下面是一个频响特性表格模型的表示形式:

```
Ehighpass     5     0     FREQ{V(10)}=
+ (10.0Hz, -  13.67, -  91.19)   (100.0Hz, 6.153, -  101.7)
+ (1.0kHz, 19.13, -  154.8)   (10.0kHz, 19.98, -  176.1)
```

其中,Ehighpass 表示一个输出电压源,其正节点号是 5,负节点是地。FREQ 表示是个频响特性的表格描述,大括号{}中的 V(10)表示输入是 10 号节点的电位,等号后面的 4 个圆括号内表示 4 组数据,每个圆括号内的三个数据分别是频率、幅度(dB)和相位(度)。这个表格模型反映了一个高通滤波器的特性。

2. 分段线性电路模型

用简单的电路元件可以很容易实现分段线性特性。我们用电阻、理想二极管和电压源构成三种支路,来实现不同的分段特性,其支路结构和对应的分段特性如图 3.6.4(a)、(b)、(c)所示,并分别称之为(a)类支路、(b)类支路和(c)类支路。

图 3.6.4　分段线性支路模型

(1) 分段线性段的斜率是递增的,即 $g_1 > g_0$,如图 3.6.4(b)所示。这时输出电压 U_O 与输入电压 U_I 之间的关系式为

$$\begin{aligned} U_O &= g_0 U_I + V_y & U_I &< V_x \\ U_O &= g_0 V_x + V_y + g_1(U_I - V_x) & U_I &\geq V_x \end{aligned} \qquad (3.6.4)$$

其中,V_x 是两个线段间连接点的横坐标值,V_y 是 g_0 与 y 轴交点的纵坐标值。而且,当 $U_I \geq V_x$ 时,有

$$\Delta U_O = (g_1 - g_0) U_I - (g_1 - g_0) V_x \qquad (3.6.5)$$

我们令电压源 $E_2 = V_x$,那么支路中电阻 R_2 与分段线段斜率 g_0, g_1 之间的关系为

$$\frac{1}{R_2} = g_1 - g_0$$

即

$$R_2 = \frac{1}{g_1 - g_0} \qquad (3.6.6)$$

(2) 分段线性段的斜率是递减的,即 $g_1 < g_0$,如图 3.6.4(c)所示。这时输出电压 U_O 与输入电压 U_I 关系式同(3.6.4)式,而当 $U_I < V_x$ 时, ΔU_O 为

$$\Delta U_O = (g_0 - g_1) U_I - (g_0 - g_1) V_x \qquad (3.6.7)$$

令支路中电压源 $E_3 = V_x$,那么支路电阻 R_3 为

$$R_3 = \frac{1}{g_0 - g_1} \qquad (3.6.8)$$

(3) 理想二极管的开关特性。在支路模型中,二极管的作用是一个压控开关。

设二极管两端电压为 U_D,电流为 I_D,则其特性公式为

$$U_D = N V_T \ln \frac{I_D}{I_S} + 1 \qquad (3.6.9)$$

其中,V_T 为热电压,I_S 是反向饱和电流,N 是发射系数。由(3.6.9)式可以看出,发射系数 N 若小于 1,则二极管上电压减小,二极管电流迅速增加。图 3.6.5 中表示了二极管反向饱和电流等于隐含值(10^{-14}A),发射系数分别等于 0.001, 1 和 2 时的 I-U 特性曲线。由此得知,当 $N = 0.001$ 时,二极管基本上可以当作一个理想开关使用。

至于图 3.6.4 中(a)类支路的作用,在下面的实例中讨论。

下面我们以图 3.6.6 所示的三段分段线性特性为例,介绍如何用上面所述的支路构造一个实际的分段线性电路宏模型。该模型由三部分电路组成:(1)输入级,(2)功能级,

图 3.6.5　具有不同发射系数的二极管特性曲线

图 3.6.6　分段特性曲线

(3)输出级。如图 3.6.7。

图 3.6.7　分段线性宏模型

由图 3.6.6 分段线性特性可得到三条线段的斜率分别为: $g_1 = -1$, $g_2 = 1$, $g_3 = -1$。

图 3.6.7 中, R_2、D_2、E_2 所在支路属于(b)类支路, 由公式(3.6.6)可得

$$R_2 = \frac{1}{g_2 - g_1} = \frac{1}{2}\Omega$$

$E_2 = -1$V, 即为 A 点横坐标, D_2 参数 $I_S = 10^{-14}$A, $N = 0.001$。

R_3、D_3、E_3 所在支路属于(c)类支路, 由公式(3.6.8)可得

$$R_3 = \frac{1}{g_2 - g_3} = \frac{1}{2}\Omega$$

$E_3 = 1$V, 即为 B 点横坐标, D_3 参数同 D_2, 也是理想二极管。

R_1、E_1 所在的(a)类支路中, R_1 和 E_1 值可通过分别计算 R_2 支路和 R_3 支路在 C 点和 D 点的电流来确定。在 C(-3, 1)点, R_3 支路中的 D_3 导通, R_2 支路中的 D_2 截止, 于是电压源 E_{TEST} 的电流为

$$I(E_{TEST}) = \frac{E_I - E_1}{R_1} - \frac{E_3 - E_I}{R_3} = 1$$

即

$$9R_1 + E_1 = -3 \qquad (3.6.10)$$

其中, $E_I = U_I$。

在 D(3, - 1)点, R_2 支路中的 D_2 导通, R_3 支路中的 D_3 截止, 于是 E_{TEST} 的电流为

$$I(E_{TEST}) = \frac{E_I - E_2}{R_2} + \frac{E_I - E_1}{R_1} = -1$$

即

$$-9R_1 + E_1 = 3 \tag{3.6.11}$$

其中, $E_I = U_I$。

由(3.6.10)和(3.6.11)式, 得

$$R_1 = -\frac{1}{3}\Omega, \quad E_1 = 0V$$

图 3.6.7 中输入级和输出级分别起阻抗匹配和电压、电流之间的转换作用。我们设输入输出级元件参数为:

$$R_I = 1M\Omega, \quad R_O = 1k\Omega, \quad R_T = 10^9\Omega$$

电压控制电压源 E_I 的作用是将输入电压 U_I 按比例转化到功能级, 其控制系数等于单位 1。电流控制电压源 H_O 的作用是将功能级电流 $I(E_{TEST})$ 转化为输出电压, 其控制系数等于单位 1。零值电压源 E_{TEST} 的作用就是取其支路电流。

3.7 模型参数提取

我们知道, 电路分析结果的精确性除了取决于计算方法的正确和计算机的字长以外, 主要依赖于电路中器件模型的精度。而模型精度本身, 又紧密地依赖于所用模型参数能否正确地反映器件的真实特性。因此, 在计算机辅助电路分析与设计中, 设计者面临着一个共同的问题就是模型参数的确定问题。晶体管的模型参数与出厂的手册参数有很大差别, 模型参数主要是半导体器件的物理参数、工艺参数, 而不是外特性参数。另外, BJT 管和 MOS 管的模型参数都有 40 个左右, 比器件手册上的参数多很多, 所以模型参数的确定是一个很棘手的问题。模型参数确定的最直接的方法是测量法。不少专门介绍半导体器件模型的书籍中都有关于模型参数测试方法和计算公式等内容。但测量法所需的测试仪器较多, 价格昂贵, 而且有些参数的测量精度较差, 有的参数甚至很难测定。因此, 模型参数的提取一般采用优化拟合的方法。这种方法是以晶体管手册参数的数据和特性曲线为初始数据, 以晶体管的工作电流、极间电容等随端电压变化的数学公式为主要计算公式, 以计算出的晶体管特性与手册中测试特性之间的误差最小为目标函数进行优化拟合, 最后提取出有关的模型参数。目前一些商用电路仿真工具中包含有模型参数提取软件包和已提出的晶体管模型参数的数据库, 供用户使用。

晶体管器件模型参数的优化提取工作主要步骤如下。

(1) 以半导体手册上查到的器件特性的数据和曲线, 或者是用户自己实际测量的器件特性的数据, 作为模型参数提取的设计依据。

(2) 建立以模型参数为变量的模型数学公式。

(3) 选择参数优化及曲线拟合的算法, 通常是采用非线性函数最小二乘法, 编制相应的优化提取软件程序。

(4) 给定模型参数的一组初始猜测值, 将其代入优化程序中的模型数学公式, 计算器

件的各种输出特性。

(5) 将计算出的特性与手册上实际测试特性进行比较,如果满足误差要求,则这组模型参数即为优化提取的参数。否则,采用适当的措施修改这组参数,返回步骤(4)再继续优化。

后面图 3.7.2 所示的是一个以 BJT 管直流模型参数提取为例的模型参数优化提取框图。

BJT 晶体管的模型参数共有 40 个,参阅表 3.2.1。其中 18 个是直流参数: I_S, β_F, β_R, n_F, n_R, c_2, c_4, I_{KF}, I_{RF}, n_{EL}, n_{CL}, U_A, U_B, R_{CC}', R_{EE}', R_{BB}', R_{BM}, I_{RB}。它们可由 I_C-U_{CE}和 I_E-U_{BE}两组特性曲线确定。交流参数有 11 个: C_{JEo}, C_{JCo}, ϕ_E, ϕ_C, m_E, m_C, C_{SUB}, ϕ_S, m_S, X_{CJC}和 FC,可由 C_{ib}-U_{EB}和 C_{ob}-U_{CB}两组特性曲线确定, C_{ib}是发射结电容, C_{ob}是集电结电容。瞬态参数有 6 个: τ_F, τ_R, $X_{\tau F}$, $U_{\tau F}$, $I_{\tau F}$, $P_{\tau F}$,可由 f_T-I_C 特性确定。此外还有 5 个与温度、噪声有关的参数: X_{TB}, X_{TI}, E_g, K_f, a_f。

下面我们以提取双极型晶体管的直流模型参数为例,介绍用最小二乘法作为优化方法进行优化提取的基本原理。

设 X 是 BJT 晶体管的直流参数矢量,我们选择 BJT 晶体管集电极电流 I_C 和基极电流 I_B 的公式为模型参数公式,它是 U_{BE}, U_{BC}及模型参数矢量 X 的函数,即

$$I_C = f(X, U_{BE}, U_{BC})$$

$$= \frac{I_S}{Q_B}\left(\exp\frac{qU_{BE}}{n_F kT} - \exp\frac{qU_{BC}}{n_R kT}\right) - \frac{I_S}{\beta_R}\left(\exp\frac{qU_{BC}}{n_R kT} - 1\right)$$

$$- c_4 I_S\left(\exp\frac{qU_{BC}}{n_{CL} kT} - 1\right) \tag{3.7.1}$$

$$I_B = \frac{I_S}{\beta_F}\left(\exp\frac{qU_{BE}}{n_F kT} - 1\right) + \frac{I_S}{\beta_R}\left(\exp\frac{qU_{BC}}{n_R kT} - 1\right)$$

$$+ I_{SE}\left(\exp\frac{qU_{BE}}{n_{EL} kT} - 1\right) + I_{SC}\left(\exp\frac{qU_{BC}}{n_{CL} kT} - 1\right) \tag{3.7.2}$$

其中, Q_B 是基区多数载流子电荷。这两个公式是 SPICE 程序中 BJT 管的 GP 模型的 I_C 和 I_B 的计算公式。

在不同偏压 U_{BE}、U_{CE}条件下,由 BJT 晶体管输出特性曲线上采样 n 个集电极电流 I_C 数据,这 n 个数据用 $\hat{I}_C^i$ $i = 1, 2, \cdots n$ 表示,如图 3.7.1 所示,图中× 为采样的数据点。

给定模型参数矢量 X 的初值,在相同偏置 U_{BE}, U_{CE}条件下,由(3.7.1)和(3.7.2)式(或 SPICE 程序)计算出集电极电流,可以表示为

$$I_C^i = f(X, U_{BE}, U_{CE}) \quad i = 1, 2, \cdots, n \tag{3.7.3}$$

最小二乘法的主要思想就是不断调整模型参数矢量 X,使 I_C^i 与实际特性值 $\hat{I}_C^i$的总的方差和最小,即求

$$\min_{X \in \Omega} \sum_{i=1}^{n} \left(I_C^i - \hat{I}_C^i\right)^2 \tag{3.7.4}$$

其中 Ω为可行域,是指晶体管直流模型参数的物理范围。这种算法实际上就是以(3.7.4)式为目标函数,用最小二乘法对模型参数进行优化。

如果我们令

$$\phi_i(X) = I_C^i - \hat{I}_C^i \qquad i = 1, 2, \cdots, n \tag{3.7.5}$$

则有

$$F(X) = \sum_{i=1}^{n} \phi_i^2(X) = \sum_{i=1}^{n} \left(I_C^i - \hat{I}_C^i\right)^2 \tag{3.7.6}$$

求 $F(X)$ 极小问题，就转化为求解下列与 I_C^i 相关的非线性方程组：

$$\begin{aligned}
&\phi_1(x_1, x_2, \cdots, x_m) = 0\\
&\phi_2(x_1, x_2, \cdots, x_m) = 0\\
&\cdots\cdots\cdots\cdots\cdots\cdots\\
&\phi_n(x_1, x_2, \cdots, x_m) = 0
\end{aligned}$$

其中 m 为模型参数矢量 X 的维数，对 BJT 晶体管直流模型参数 m= 18。由此求解出极小点 X^* 即为模型参数优化值。这个过程称为晶体管模型参数的优化提取。

图 3.7.1　BJT 晶体管输出特性曲线的测量数据

图 3.7.2　模型参数优化提取框图

模型参数优化提取的框图如图 3.7.2 所示。

在图 3.7.1 中，实线为优化提取的模型参数所计算出的晶体管输出特性曲线。可以看出，计算曲线与测量曲线相符，说明优化提取的模型参数是可用的。我们通过模型参数提取工作能建立一个常用半导体器件模型参数数据库，将给电路 CAD 工作带来很大方便，并确保了模拟的精度。

习　　题

3.1　用 PSpice 程序计算题图 3.1 所示 BJT 管的输入特性曲线。输入数据文件为

```
BJT   INPUT   CHARACTER
Q1   2   1   0   MOD      ;题图 3.1 中为 T1
```

```
VBE   1   0
VCE   2   0
.MODEL   MOD   NPN   IS= 1E- 15   BF= 80
+ VAF= 100   RC= 100   RB= 20
.DC   VBE   0   2   0.05   VCE   0   5   1
.PROBE
.END
```

用 PSpice 的绘图软件 PROBE 绘出 I_B-U_{BE}曲线。

题图 3.1

3.2 用 PSpice 程序计算题图 3.2 所示 BJT 管的输出特性曲线。设晶体管参数为

$$I_S = 10^{-15}\text{A},\ \beta_F = 100,\ U_A = 100\text{V}$$

(1) I_B 从 0～40 μA 变化, ΔI_B= 10 μA, 同时, U_{CC}从 0～10V 变化, ΔU_{CC}= 0.2V, 计算输出特性曲线 I_C-U_{CE}。

(2) 比较模型参数 U_A 分别等于 100V 和 50V 时, 输出特性曲线的变化。

(3) 管子集电极体电阻 R_{CC}= 0.1Ω, 100Ω, 1kΩ时, 观察输出特性曲线的变化。

题图 3.2

题图 3.3

3.3 用 PSpice 程序计算题图 3.3 所示 MOS 场效应晶体管的输出特性曲线。MOS 管的 PSpice 参数如下:

```
L= 4U   W= 6U   AD= 10P
AS= 10P   VTO= - 2V
NSUB= 1E15   UO= 550
```

第4章　直流分析

电路的直流分析是在电路中电容开路、电感短路的情况下，计算电路的静态工作点，即在恒定激励条件下求电路的稳态解。在电子线路中，无论是大信号还是小信号工作状态，都必须给半导体器件以正确的偏置，以便使其工作在所需的区域，这就是直流分析问题。例如，在交流小信号分析中，电路先要进行直流分析，以确定半导体器件的跨导等小信号参数；在瞬态分析中，需求出电路在指定时间区间上的解，这时的电路方程是个常微分方程，求解常微分方程必须先求出电路储能元件上的初始电流或电压值，这也由直流分析来完成。所以，电路的直流分析是电路分析的基础。

当电路中某些元器件的特性方程是电路变量的非线性函数时，我们称这个电路为非线性电路。在电路领域中，除了少数无源网络属于线性电路外，绝大多数电路都是非线性的。组成非线性电路的非线性元器件有：二极管，双极型晶体管，MOS 场效应管等半导体器件，磁性材料器件，光电器件，以及非线性电阻、电容、电感，非线性受控源等等。其中半导体器件的模型的特性方程我们在前面第 3 章中已做了介绍。本章将着重介绍如何在直流分析中处理和求解含有半导体器件的非线性电路问题。

线性电路的直流分析，所建立的电路方程是线性代数方程组；非线性电路的直流分析，所建立的方程是非线性代数方程组。线性代数方程组的求解方法，我们在第 2 章中已做了介绍。非线性方程的求解则要复杂得多，通常采用数值迭代算法，将非线性方程转化为线性代数方程，即确定一个线性代数方程组的解序列，然后用线性代数方程的求解方法，如 LU 分解法或高斯消元法在每个迭代点上求得线性方程的解，直到迭代收敛。

在本章中，我们着重介绍非线性数值解法中应用最广的牛顿-拉夫森(Newton-Raphson 方法，简称 N-R 方法)，以及牛顿-拉夫森方法的几种改进算法。非线性迭代算法的收敛性问题，是一个长期没能得到很好解决的难题，到目前为止还没有找出统一的能适用于所有非线性电路的求解方法。本章中，我们对目前国际上流行的一些改善收敛的算法也进行简单的介绍。

4.1　线性直流分析

在线性直流分析中，电路的所有元器件模型都是线性模型，其特性方程也是线性方程，故最后形成的电路方程组是个线性代数方程组。通常用改进节点法列电路方程，用 LU 分解法求解电路方程组，电路方程组的解就是电路的直流工作点。

直流线性分析时，常用的电路元器件有：线性电阻、独立源、线性受控源等等。这些线性元器件的模型和特性方程我们在第 2 章 2.1 节中已做了介绍，读者可以从表 2.1.7 和表 2.1.10 中查到这些元器件的模型以及它们对电路方程组的贡献。直流分析时，电路中的电容开路，电感短路。为了避免由于电容开路和电感短路而改变电路的拓扑结构，通常

可将电容设为一个小电导,如 $G_C = 10^{-7}S$;电感设为一个大电导,如 $G_L = 10S$。电路中的二极管,晶体三极管等非线性器件可用独立源或受控源来近似,但这种近似是很粗略的。

4.1.1 直流分析功能

我们以 MicroSim 公司的 PSpice 程序中的直流分析部分为例来介绍直流分析功能。

1. 确定电路的直流工作点

这是直流分析的主要功能,计算时电路中的电容开路(晶体管的极间电容也不考虑),电感短路。在进行瞬态分析之前,PSpice 程序自动先进行直流分析,以决定瞬态时电路的初始条件。同样,在进行交流小信号分析之前,也先自动进行直流分析,以决定非线性半导体器件的线性化模型参数。

PSpice 中计算直流工作点的语句是 .OP。

2. 计算电路的直流传输曲线

这是在指定的范围内计算电路直流输出变量与某个(或某两个)独立电压源或独立电流源步进变化之间的对应关系曲线。PSpice 中计算直流传输曲线的典型语句是

```
.DC   VCE   0   10   0.25   IB   0   40U   5U
```

这个语句是使电压源 VCE 的值从 0 扫描到 10V,每次增量为 0.25V;电流源 IB 的值从 0 扫描到 40 μA,每次增量为 5 μA。可以通过绘图观测到电路中任何一个节点的电压(或任一支路电流)随这两个独立源变化的特性曲线。

3. 计算直流传输函数

这是通过直流分析确定电路的输出和输入的关系,即输出变量对输入变量的传输函数,包括电路的直流输入和输出电阻。PSpice 中计算直流传输函数的语句实例是

```
.TF   V(5, 3)   VIN
```

其中 VIN 是输入电压源的变量名。针对这条语句,PSpice 将计算 $\frac{V(5)-V(3)}{VIN}$ 的值,以及 VIN 端的输入电阻,节点(5)与节点(3)之间的输出电阻。

直流分析的上述三种功能,无论对线性网络还是非线性电路都是适用的。

4.1.2 直流线性分析流程

在商用电路 CAD 程序中一般设有专门的线性直流分析部分,通常线性与非线性分析是联成一体的。为了便于读者理解,我们先单独介绍直流线性部分。

一个直流线性分析程序主要由以下几部分组成:

(1) 读入电路的输入数据

电路输入数据是一个描述电路拓扑结构,元器件参数,分析电路的条件及输出形式,要求等的一个数据文件。它实际上是由某种电路描述语言来实现的,这种电路描述语言既要符合用户的习惯,便于掌握,又要易于计算机编译和处理。不同的电路分析程序,其电路描述语言是不相同的,前面电路分析实例中电路输入数据文件,就是用 SPICE 程序要求的电路描述语言构成的。目前,不少商用的电路 CAD 系统采用绘制电路图的方式作为电路数据的输入手段,当用户在计算机上绘制好电路图后,计算机能自动地将图形数据转换

成电路 CAD 程序所能接受的数据形式。

由上所述,读入电路输入数据程序主要是一个编译程序,它将用户的电路图形式或电路描述文件形式的输入数据编译成电路分析中建立的元器件数学模型和电路方程所能识别的内部数据。这个读入输入数据的程序称为人机交互界面,它是用户和电路分析程序之间的桥梁。

(2) 建立器件模型和电路方程

在不同的分析领域中,电路元器件模型可能是不相同的。比如电容,它在直流、交流小信号及瞬态分析中呈现三种不同的模型形式。第 2 章 2.1 节中介绍的各种器件模型,大多是线性稳态分析的器件模型。这部分程序以这些器件模型为依据编制程序,将各元器件对电路方程的贡献填入到方程的系数矩阵和右端向量中去,并最后形成电路方程。

(3) 求解电路方程组

电路 CAD 中常用的解方程算法是 LU 分解法,它是一个通用的解方程程序。电路 CAD 中通常采用稀疏矩阵技术求解方程,以提高求解速度和效率。

(4) 输出数据和图表

直流线性分析结果是电路的静态节点电位或支路电流信息,传输特性曲线,直流传输函数值等等。

处理电路分析输出结果数据的程序,我们常称之为输出后处理程序。它不但能输出表格形式的电压、电流、功率等数据信息,还能以图形形式绘制出输出曲线或波形,甚至可以在分析程序执行过程中,实时地跟踪记录并绘制电路输出特性曲线或波形。

直流线性分析程序的流程如图 4.1.1 所示。

图 4.1.1　直流线性分析流程图

4.1.3　直流线性分析实例

例 4.1.1　计算图 4.1.2 所示线性网络的直流工作点。

电路输入数据文件如下:

```
Simple linear network
R1   1   2   2
R2   3   0   2
```

图 4.1.2　线性网络

```
R3   2   6   2
R4   4   0   2
R5   4   5   2
VE   1   0   12 ; 图 4.1.2 中的 E1
I1   4   2   1
VB   6   4   0 ; 图 4.1.2 中的 E64
G1   5   0   3   0   0.1
H1   2   3   VB   0.2
.OP
.END
```

数据文件中, VE 为恒压源; I1 为恒流源; G1 为 VCCS, 它是由节点⑤接到地的一个受控电流源, 控制电压是节点③到地的电压 V_3, 控制系数为 $g_m = 0.1S$; H1 为 CCVS, 是由正节点②到负节点③的一个受控电压源, 其控制电流是由节点⑥到节点④的一个零值电压源 E_{64} 提供的, 控制系数是阻抗 $r = 0.2\Omega$。

由 PSpice 的直流分析计算出的该电路的静态工作点如下:

NODE	VOLTAGE	NODE	VOLTAGE	NODE	VOLTAGE	NODE	VOLTAGE
(1)	12.0000	(2)	5.1719	(3)	4.7656	(4)	1.1094
(5)	0.1563	(6)	1.1094				

例 4.1.2　图 4.1.3 是一个运算放大器的直流宏模型电路, 按照设计指标它的差模开环增益是 2×10^5, 差模输入电阻是 2MΩ, 输出电阻是 75Ω。我们用直流传输函数计算此电路的增益和输入输出电阻, 验证该模型的正确性。

电路输入数据文件如下:

```
OP   DC   MacroModel
RC   1   3   100
RID   2   4   2MEG
RIC+   2   0   2E+ 9
```

图 4.1.3　运算放大器直流宏模型

```
RIC-    4   0   2E+9
RO   5   0   75
VIO   4   3   2M                    ;图中的 E_IO
I+    2   0   8E-8
I-    4   0   9E-8
GM2   4   1   2   0   .316E-6   ;图中的 g_m2
GM1   0   5   2   4   2.67E+3   ;图中的 g_m1
VIN   1   2   1                     ;图中的 U_I
.TF   V(5)   VIN
.END
```

计算结果如下:

```
SMALL-SIGNAL   CHARACTERISTICS
V(5)/VIN =    -2.002E+05
INPUT   RESISTANCE   AT   VIN =    1.999E+06
OUTPUT   RESISTANCE   AT   V(5) =    7.500E+01
```

可以看出计算结果与原设计是一致的。

若用此运放做一个加法器,如图 4.1.4 所示,两个输入电压分别为 1V 和 2V,计算加法器输出电压值。

为此,我们先将该运放宏模型定义为一个子电路,即

```
.SUBCKT   OPA   1   2   5
                  |   |   └─ out
                  |   └──── in-
                  └──────── in+
```

整个加法器的输入数据文件如下:

```
The   Adder   Amp.
R1   1   3   10K
R2   2   3   10K
RF   3   4   10K
XOP   0   3   4   OPA
```

图 4.1.4　加法器

```
VI1  1  0  DC  1.0          ;图 4.1.4 中的 UI1
+   PULSE(0  1  0  0  0  2E- 4  4E- 4)
VI2  2  0  DC  2.0          ;图 4.1.4 中的 UI2
+   PULSE(0  1  1E- 4  0  0  2E- 4  4E- 4)
* * *   Opam  subckt  * * *
.SUBCKT  OPA  1  2  5
RC  1  3  100
RID  2  4  2MEG
RIC+   2  0  2E+ 9
RIC-   4  0  2E+ 9
R0  5  0  75
VIO  4  3  2M          ;图 4.1.3 中的 EIO
I+   2  0  8E- 8
I-   4  0  9E- 8
GM1  4  1  2  0  .316E- 6
GM2  0  5  2  4  2.67E+ 3
.ENDS
* * * * * * * * * * * * * * *
.OP
.TRAN  5E- 6  1E- 3
.PROBE
.END
```

输入文件中,XOP 为调用运放宏模型子电路的语句。该电路用 PSpice 计算出的静态工作点为

NODE	VOLTAGE	NODE	VOLTAGE	NODE	VOLTAGE
(1)	1.0000	(2)	2.0000	(3)	34.53E-6
(4)	-2.9991				

输出节点(4)的电位为-2.9991V,是输入 U_{I1} 与输入 U_{I2} 的和(反相)。

瞬态分析结果如图 4.1.5 所示。

图 4.1.5　加法器瞬态分析结果

4.2 非线性直流分析的数值方法

当电路中含有非线性器件时,电路方程是一个非线性方程。例如图 4.2.1 所示的一个简单二极管电路,其支路特性方程可以写成如下形式:

$$I_R = \frac{E - U_D}{R} \qquad (4.2.1)$$

$$I_D = I_S\ e^{U_D/V_T} - 1 \qquad (4.2.2)$$

其中,二极管的特性方程(4.2.2)是一个非线性方程,$V_T = \frac{nkT}{q}$。将(4.2.1)和(4.2.2)式合并,可表示为

$$f(U_D) = \frac{E - U_D}{R} - I_S\ e^{U_D/V_T} - 1 = 0 \qquad (4.2.3)$$

图 4.2.1 简单二极管电路

这就是以二极管两端电压 U_D 为变量的非线性电路方程。

如何求解非线性电路方程,这是我们这一章要介绍的主要内容。我们先从数学的角度去论述如何用数值迭代方法求解非线性方程,然后结合半导体器件的伴随模型,阐述它在电路中的应用。

4.2.1 简单迭代法

简单迭代法是求解非线性方程的最原始的方法。

设非线性代数方程组为

$$F(X) = 0 \qquad (4.2.4)$$

这里 X 是电路的未知变量向量。首先将(4.2.4)式化为 $X = F_1(X)$ 的形式,然后设计一个迭代公式:

$$X^{k+1} = F_1(X^k) \qquad k = 0, 1, 2, \cdots \qquad (4.2.5)$$

选定一个解的初始猜测值 X^0,代入(4.2.5)式,产生迭代序列 X^{k+1},当迭代到

$$\| X^{k+1} - X^k \| \leq \varepsilon$$

(其中 ε 为给定误差限),则认为 X^{k+1} 即为方程(4.2.5)的解。

我们称 $F_1(X)$ 为迭代函数,产生的序列 X^{k+1} 为迭代序列,ε 为给定误差。迭代法所涉及的基本问题是:迭代函数 $F_1(X)$ 的构造,迭代序列 X^{k+1} 的收敛性、收敛速度和迭代的误差估计。

我们以一个方程为例,说明简单迭代法的计算过程。

例 4.2.1 用简单迭代法求解方程 $4 - x - 2x^{1/3} = 0$。

解 设迭代函数为

$$x = 4 - 2x^{1/3}$$

取 $x^0 = 2$ 作为迭代初值,则

$$x^1 = 4 - 2 \times 2^{1/3} = 1.4801$$

$$x^{2} = 4 - 2\times 1.4801^{1/3} = 1.7207$$

……

$$x^{10} = 4 - 2\times 1.6405^{1/3} = 1.6412$$

$$x^{11} = 4 - 2\times 1.6412^{1/3} = 1.6409$$

$$x^{12} = 4 - 2\times 1.6409^{1/3} = 1.6410$$

$$x^{13} = 4 - 2\times 1.6410^{1/3} = 1.6410$$

设误差 $\varepsilon = 10^{-4}$, 显然

$$\| x^{13} - x^{12} \| < \varepsilon$$

故 $x^{13} = 1.6410$ 为方程的解, 迭代次数 13。我们用图 4.2.2 来表示上述迭代过程, $y = x$ 与 $y = 4 - 2x^{1/3}$ 的交点就是 $4 - x - 2x^{1/3} = 0$ 的解。

简单迭代法虽然简单, 但它很容易使迭代发散。为了保证迭代过程收敛, 要求迭代函数 $F_1(X)$ 在域内连续, 且其导数绝对值小于 1, 即 $|F_1'(X)| < 1$, 这往往是很困难的。而且即使能够收敛, 收敛速度也很慢。可以证明当简单迭代法收敛时, 它是以一次方速度收敛, 即线性收敛。

图 4.2.2 简单迭代法示意图

设 $\delta X^k = X^{k+1} - X^k$, 由式(4.2.5)有

$$\begin{aligned}\delta X^k &= F_1(X^k) - F_1(X^{k-1}) \\ &= F_1'(\xi)(X^k - X^{k-1}) \\ &= F_1'(\xi)\delta X^{k-1}\end{aligned}$$

其中, $$X^{k-1} \leq \xi \leq X^k$$

只有当 $|F_1'(X)| < 1$ 时, $|\delta X^k| < |\delta X^{k-1}|$, 当 $k \to \infty$ 时, $\delta X^k \to 0$, 故当 $|F_1'(X)| < 1$ 时, δX^k 以一次方的速度减小。

4.2.2 牛顿-拉夫森方法

牛顿-拉夫森(Newton-Raphson)方法是求解非线性方程中应用最广泛、也最有效的一种方法。它是目前多数电路分析程序中非线性分析方法的基础。

设 X^k 是非线性方程 $F(X) = 0$ 的第 k 次迭代解, 则第 $k+1$ 次迭代解为

$$X^{k+1} = X^k + \delta X^k \tag{4.2.6}$$

当 δX^k 很小时, 将 $F(X^{k+1})$ 在 X^k 处展开成台劳级数, 并略去高次项, 则有

$$F(X^{k+1}) = F(X^k + \delta X^k) \approx F(X^k) + J(X^k)\delta X^k \tag{4.2.7}$$

这就是 N-R 方法的迭代函数。当 $F(X) = 0$ 是单个方程时, 有

$$J(x^k) = F'(x^k)$$

当 $F(X) = 0$ 是 n 阶方程组, 即

$$F(X) = [f_1(X), f_2(X), \cdots, f_n(X)]^T \tag{4.2.8}$$

时, $J(X^k)$ 是雅可比(Jacobian)矩阵, 其定义为

$$J\left(X^{k}\right)=\begin{bmatrix}\left.\frac{f_1}{x_1}\right|_{X^k} & \left.\frac{f_1}{x_2}\right|_{X^k} & \cdots & \left.\frac{f_1}{x_n}\right|_{X^k}\\ \left.\frac{f_2}{x_1}\right|_{X^k} & \left.\frac{f_2}{x_2}\right|_{X^k} & \cdots & \left.\frac{f_2}{x_n}\right|_{X^k}\\ \vdots & \vdots & & \vdots\\ \left.\frac{f_n}{x_1}\right|_{X^k} & \left.\frac{f_n}{x_2}\right|_{X^k} & \cdots & \left.\frac{f_n}{x_n}\right|_{X^k}\end{bmatrix} \tag{4.2.9}$$

若 X^{k+1} 为方程 $F(X)=0$ 的第 $k+1$ 次近似解，则

$$F\left(X^{k+1}\right)\approx 0$$

故由(4.2.7)式，有

$$F\left(X^{k+1}\right)=F\left(X^{k}\right)+J\left(X^{k}\right)\delta X^{k}=0 \tag{4.2.10}$$

将(4.2.6)式代入(4.2.10)式，得

$$J\left(X^{k}\right)\left(X^{k+1}\right)=-F\left(X^{k}\right)+J\left(X^{k}\right)X^{k} \tag{4.2.11}$$

这就是 N-R 法的迭代关系式，它是一个线性代数方程组。其中 X^{k+1} 是未知变量向量。当 X^{k} 已知时，$J\left(X^{k}\right)$ 是方程系数矩阵。这个方程组可以用第 3 章介绍的高斯消去法或 LU 分解法求解。

当给定一个恰当的初值 X^{0} 以后，代入(4.2.11)式进行迭代，产生一个迭代序列 X^{k+1}，直到相邻两次的解向量 X^{k+1} 与 X^{k} 之间差的绝对值小于某个给定的允许误差 ε 为止。这就是经典的 N-R 方法。

N-R 迭代过程的几何解释如图 4.2.3 所示。首次迭代时 $k=0$，$x^{k}=x^{0}$ 即为迭代初值，由 x^{0} 求出 x^{1}，再由 x^{1} 求出 x^{2}，……这样得到的一个序列 x^{k+1}，最后收敛于真值 x^{*}。从图 4.2.3 还可以看出，N-R 方法相当于在每个迭代点 $x^{k}, f\left(x^{k}\right)$ 上做切线，切线交于 x 轴得到 x^{k+1}；再经过 $x^{k+1}, f\left(x^{k+1}\right)$ 点做切线，与 x 轴相交得 x^{k+2}，……如此继续直到逼近真值 x^{*}。

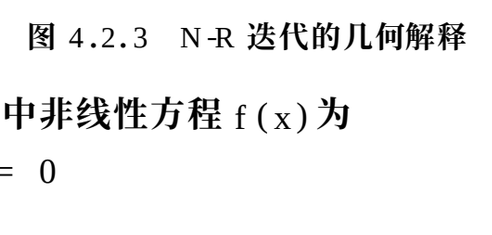


图 4.2.3 N-R 迭代的几何解释

非线性求解方法的计算速度取决于迭代序列收敛的迭代次数。可以证明，如果 X^{k} 充分接近精确解，则有

$$\delta X^{k+1}=C\left(\delta X^{k}\right)^{2} \tag{4.2.12}$$

其中系数 C 取决于方程的高阶导数。也就是说，N-R 方法具有二次方的收敛速度，即每迭代一次，误差以平方的速度减小。

我们仍以例 4.2.1 来看 N-R 方法的收敛情况。例中非线性方程 $f(x)$ 为

$$f(x)=x-4+2x^{1/3}=0$$

$f(x)$ 的一阶导数为

$$f'(x)=1+\frac{2}{3}x^{-2/3}$$

N-R 迭代关系式为

$$x^{k+1} = x^k - \frac{f(x^k)}{f'(x^k)}$$

设初始值 $x^0 = 2$, 则

$$x^1 = x^0 - \frac{f(x^0)}{f'(x^0)} = 2 - \frac{2 - 4 + 2\times 2^{1/3}}{1 + \frac{2}{3}\times 2^{-2/3}} = 1.6339$$

$$x^2 = x^1 - \frac{f(x^1)}{f'(x^1)} = 1.6410$$

$$x^3 = x^2 - \frac{f(x^2)}{f'(x^2)} = 1.6410$$

设 $\varepsilon = 10^{-4}$, 显然, $|x^3 - x^2| < \varepsilon$ 故 $x^3 = 1.6410$ 为方程的解。由此例可知, N-R 方法只用了三次就收敛了, 而前面所述简单迭代法却要用 13 次迭代才收敛。可见 N-R 方法收敛速度快且有效。

N-R 迭代算法的步骤如下:

(1) 构造非线性方程 $F(X) = 0$ 的 N-R 迭代关系式。

$$X^{k+1} = X^k - [J(X^k)]^{-1} F(X^k) \qquad (4.2.13)$$

(2) 将 X^k 代入 N-R 关系式(4.2.13)进行迭代, 首次 $k = 0$, 即 X^0 为给定迭代初值, 计算出 X^{k+1}。

(3) 在给定允许误差 ε的情况下, 判断 X^{k+1}与 X^k 之间是否满足 $\| X^{k+1} - X^k \| < \varepsilon$若满足, 则 X^{k+1}即为非线性方程的解; 否则 $k = k + 1$, 返回(2)继续迭代。

4.3 非线性器件的直流伴随模型

N-R 方法的基本思想是在每一个迭代点上将非线性方程线性化, 即用迭代点上的切线近似非线性特性, 把非线性方程转化为各个迭代点上的线性方程。也就是说, 在每一个迭代点上非线性器件都由近似的线性化模型所代替, 只要我们分别建立起各个非线性器件的这种 N-R 线性化模型, 就可以按照第 2 章介绍的列方程方法建立起整个电路的迭代方程。要特别指出的是, 在不同的迭代点, 切线的斜率不同, 因而所形成的非线性元件的线性化模型的参数也就不同, 即线性化模型的参数是随迭代过程而变化的, 因此, 我们称之为直流伴随模型。下面我们推导二极管、双极型晶体管和 MOS 场效应晶体管的直流伴随模型, 它们是以第 3 章中介绍的基本模型为基础的。

4.3.1 二极管直流伴随模型

一个理想二极管的模型如图 4.3.1 所示, 其 PN 结的非线性特性方程为

$$I_D = I_S\left(e^{U_D/V_T} - 1\right) \qquad (4.3.1)$$

其中, V_T —— 热电压, $V_T = kT/q$。

我们将二极管特性方程(4.3.1)在第 k 次迭代点 U_D^k 附近展成台劳级数, 略去高次项, 得

$$I_D^{k+1} \approx I_S\left(e^{U_D^k/V_T} - 1\right) + G_{DM}^k\left(U_D^{k+1} - U_D^k\right) = I_{DM}^k + G_{DM}^k U_D^{k+1} \qquad (4.3.2)$$

图 4.3.1　理想二极管模型　　图 4.3.2　理想二极管直流伴随模型

其中

$$G_{DM}^{k} = \left.\frac{I_D}{U_D}\right|_{U_D = U_D^k} = \frac{I_S}{V_T} e^{U_D^k / V_T} \tag{4.3.3}$$

$$I_{DM}^{k} = I_S \left(e^{U_D^k / V_T} - 1 \right) - G_{DM}^{k} U_D^k \tag{4.3.4}$$

因此，二极管的非线性支路在第 k 次迭代点，可以用一个电导 G_{DM}^{k} 和一个电流源 I_{DM}^{k} 相并联来表示，如图 4.3.2 所示。这就是二极管的直流伴随模型。二极管直流伴随模型的几何解释如图 4.3.3 所示。

图 4.3.3　二极管直流伴随模型的几何解释　　图 4.3.4　二极管 EM 直流伴随模型

实际应用的二极管直流模型比上述模型多一个体电阻 R_s，如图 4.3.4 所示，称之为二极管 EM 直流伴随模型。由于 R_s 的存在，模型中增加了一个内节点 a′。

有了二极管直流伴随模型之后，就可以按照第 2 章介绍的节点法和改进节点法直接建立电路方程。二极管对电路方程的贡献，由送值表 4.3.1 形式表示出来。

表 4.3.1　二极管直流伴随模型送值表

行 \ 列	V_a	$V_{a'}$	V_c	RHS
a	$1/R_s$	$-1/R_s$		
a′	$-1/R_s$	$1/R_s + G_{DM}^{k}$	$-G_{DM}^{k}$	$-I_{DM}^{k}$
c		$-G_{DM}^{k}$	G_{DM}^{k}	I_{DM}^{k}

下面我们以一个简单的二极管电路为例，介绍建立二极管直流伴随模型和用此模型求解电路方程的 N-R 迭代过程。

例 4.3.1　简单二极管电路如图 4.3.5 所示，电路中二极管模型采用 EM 模型，用

N-R 方法求此电路的直流工作点。

图 4.3.5　例 4.3.1 简单二极管电路

图 4.3.6　例 4.3.1 电路的第 k 次迭代等效电路

(1) 选取迭代初值 U_D^k, 首次 k= 0, 代入直流伴随模型参数公式(4.3.3)和(4.3.4)中, 计算出 G_{DM}^k 和 I_{DM}^k, 即有

$$G_{DM}^k = \frac{I_S}{V_T} e^{U_D^k / V_T}$$

$$I_{DM}^k = I_D^k - G_{DM}^k U_D^k$$

此时电路等效为图 4.3.6 所示形式。

(2) 建立第 k+ 1 次迭代的线性代数方程组 $T^k X^{k+1} = B^k$, 即

$$\begin{bmatrix} \frac{1}{R} & -\frac{1}{R} & 0 & 1 \\ -\frac{1}{R} & \frac{1}{R}+\frac{1}{R_s} & -\frac{1}{R_s} & 0 \\ 0 & -\frac{1}{R_s} & \frac{1}{R_s}+G_{DM}^k & 0 \\ 1 & 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} V_1^{k+1} \\ V_2^{k+1} \\ V_3^{k+1} \\ I_E^{k+1} \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ -I_{DM}^k \\ E \end{bmatrix}$$

(3) 用 LU 分解法求解方程组, 求得

$$X^{k+1} = \left[V_1^{k+1}, V_2^{k+1}, V_3^{k+1}, I_E^{k+1} \right]^T$$

(4) 判断 $\| X^{k+1} - X^k \| \leq \varepsilon$ 是否成立, 若成立, X^{k+1} 即为电路的直流解; 否则 k= k+ 1, 转步骤(1)继续迭代。

图 4.3.7 表示了此例题的 N-R 迭代过程。

4.3.2　双极型晶体管的直流伴随模型

在第 3 章中介绍了各种不同复杂程度的双极型晶体管模型。在直流分析中最常用的是混合 π型 EM1 和 EM2 模型。我们以混合 π型 EM1 模型为例, 推导其 N-R 法的直流伴随模型。

图 4.3.8 是 NPN 型双极晶体管的混合 π型 EM1 模型。其 3 个非线性特性方程是

$$I_{RC} = I_S \left(e^{U_{BC}/V_T} - 1 \right) \tag{4.3.5}$$

$$I_{FC} = I_S \left(e^{U_{BE}/V_T} - 1 \right) \tag{4.3.6}$$

$$I_{CT} = I_{FC} - I_{RC} \tag{4.3.7}$$

与二极管的处理方法类似, 我们把(4.3.5)和(4.3.6)两个非线性方程在第 k 次迭代

图 4.3.7　简单二极管电路的 N-R 迭代过程示意图

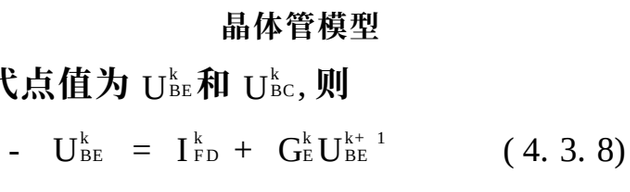


图 4.3.8　混合 π型 EM1 双极晶体管模型

点附近做台劳级数展开,略去高次项。第 k 次迭代点值为 U_{BE}^{k}和 U_{BC}^{k},则

$$I_{FC}^{k+1} \approx I_S\left(e^{U_{BE}^{k}/V_T}-1\right)+G_E^{k}\left(U_{BE}^{k+1}-U_{BE}^{k}\right)=I_{FD}^{k}+G_E^{k}U_{BE}^{k+1} \tag{4.3.8}$$

$$I_{RC}^{k+1} \approx I_S\left(e^{U_{BC}^{k}/V_T}-1\right)+G_C^{k}\left(U_{BC}^{k+1}-U_{BC}^{k}\right)=I_{RD}^{k}+G_C^{k}U_{BC}^{k+1} \tag{4.3.9}$$

其中

$$G_E^{k}=\left.\frac{I_{FC}}{U_{BE}}\right|_{U_{BE}=U_{BE}^{k}}=\frac{I_S}{V_T}e^{U_{BE}^{k}/V_T} \tag{4.3.10}$$

$$I_{FD}^{k}=I_S\left(e^{U_{BE}^{k}/V_T}-1\right)-G_E^{k}U_{BE}^{k} \tag{4.3.11}$$

$$G_C^{k}=\left.\frac{I_{RC}}{U_{BC}}\right|_{U_{BC}=U_{BC}^{k}}=\frac{I_S}{V_T}e^{U_{BC}^{k}/V_T} \tag{4.3.12}$$

$$I_{RD}^{k}=I_S\left(e^{U_{BC}^{k}/V_T}-1\right)-G_C^{k}U_{BC}^{k} \tag{4.3.13}$$

因而,BE 支路的二极管非线性特性等效为一个电导 G_E^{k} 和电流源 I_{FD}^{k}并联;同理,BC 支路等效为电导 G_C^{k} 和电流源 I_{RD}^{k}并联。对于 CE 支路,

$$I_{CT}^{k+1}=I_{FC}^{k+1}-I_{RC}^{k+1}=I_{FD}^{k}-I_{RD}^{k}+G_E^{k}U_{BE}^{k+1}-G_C^{k}U_{BC}^{k+1}$$

因而,CE 支路特性等效为一个电流源 $I_{FD}^{k}-I_{RD}^{k}$ 和两个受控源。

混合 π型 EM1 双极晶体管的 N-R 直流伴随模型为图 4.3.9 所示形式。这个直流伴随模型对电路方程组的贡献可由表 4.3.2 所示送值表描述。

表 4.3.2　EM1 双极晶体管直流伴随模型送值表

行＼列	U_C	U_B	U_E	RHS
C	$G_C^{k}+\frac{G_C^{k}}{\beta_R}$	$G_E^{k}-G_C^{k}-\frac{G_C^{k}}{\beta_R}$	$-G_E^{k}$	$I_{RD}^{k}+\frac{I_{RD}^{k}}{\beta_R}-I_{FD}^{k}$
B	$-\frac{G_C^{k}}{\beta_R}$	$\frac{G_E^{k}}{\beta_F}+\frac{G_C^{k}}{\beta_R}$	$-\frac{G_E^{k}}{\beta_F}$	$-\frac{I_{RD}^{k}}{\beta_R}-\frac{I_{FD}^{k}}{\beta_F}$
E	$-G_C^{k}$	$G_C-G_E^{k}-\frac{G_E^{k}}{\beta_F}$	$G_E^{k}+\frac{G_E^{k}}{\beta_F}$	$I_{FD}^{k}+\frac{I_{FD}^{k}}{\beta_F}-I_{RD}^{k}$

实际常用的 EM2 双极晶体管直流模型比 EM1 模型多 3 个欧姆电阻:$R_{BB'}$,$R_{EE'}$和 $R_{CC'}$,其它模型参数定义与 EM1 直流模型相同。

图 4.3.9　双极型晶体管混合 π型 EM1 直流伴随模型

4.3.3　MOS 场效应晶体管的直流伴随模型

第 3 章我们介绍了 MOS 场效应晶体管模型的基本形式, N 沟道 MOS 场效应管的直流模型如图 4.3.10 所示。该模型中两个二极管的直流线性化模型如前述理想二极管直流伴随模型一样, 可等效为一个电导和一个电流源并联。

MOS 晶体管的漏源电流 I_{DS} 是端电压 U_{DS}, U_{GS} 和 U_{SB} 的函数, 即 $I_{DS}= f(U_{DS}, U_{GS}, U_{SB})$。这是一个非线性方程。用 N-R 方法将此方程线性化, 得到 I_{DS} 的线性表达式为

$$\begin{aligned} I_{DS}^{k+1} &= I_{DSo}^{k} + \frac{I_{DS}}{U_{DS}} U_{DS}^{k+1} + \frac{I_{DS}}{U_{GS}} U_{GS}^{k+1} + \frac{I_{DS}}{U_{SB}} U_{SB}^{k+1} \\ &= I_{DSo}^{k} + G_G^{k} U_{DS}^{k+1} + g_m^{k} U_{GS}^{k+1} + g_{mB}^{k} U_{SB}^{k+1} \end{aligned} \tag{4.3.14}$$

由于 MOS 晶体管工作在不同区域时, 其电流表达式是不同的, 因而等效电导 G_G 和互导 g_m, g_{mB} 以及电流源 I_{DCo} 的取值也不相同。

(1) 截止区

$$G_G = g_m = g_{mB} = 0$$

(2) 线性区

$$G_G = \beta(U_{GS} - V_{TH} - U_{DS}) \tag{4.3.15}$$

$$g_m = \beta U_{DS} \tag{4.3.16}$$

$$g_{mB} = -\frac{g_m}{2\sqrt{2\phi - U_{BS}}} \tag{4.3.17}$$

(3) 饱和区

$$G_G = \lambda I_{DS}$$

$$g_m = \frac{2I_{DS}}{U_{GS} - V_{TH}} \tag{4.3.18}$$

$$g_{mB} = -\frac{g_m}{2\sqrt{2\phi - U_{BS}}} \tag{4.3.19}$$

等效电流源 I_{DSo} 由前一次迭代结果决定，因而有

$$I_{DSo}^{k} = I_{DS}^{k} - G_{G}^{k}U_{DS}^{k} - g_{m}^{k}U_{GS}^{k} - g_{mB}^{k}U_{SB}^{k} \qquad (4.3.20)$$

以上各公式中的参数意义与第 3 章 MOS 晶体管模型的基本形式中定义的相同。由以上公式推导所得到的 MOS 晶体管直流伴随模型如图 4.3.11。

图 4.3.10　MOS 晶体管直流模型

图 4.3.11　MOS 晶体管直流伴随模型

4.4　N-R 方法的收敛性

前面已经提到，N-R 方法具有平方收敛速度，在各种迭代算法中属于收敛较快，因而也应用较为广泛的一种非线性迭代算法。

N-R 算法的收敛判据通常是要求前后两次迭代的解之差在一个指定的误差范围之内，这个收敛判据可以写成如下不等式形式：

$$\left|X_{i}^{k+1} - X^{k}\right| < \varepsilon + \varepsilon\min\{\left|X_{i}^{k}\right|, \left|X_{i}^{k+1}\right|\} \qquad (4.4.1)$$

其中，ε 是绝对误差限，ε 是相对误差限。在改进节点法形成的混合节点方程的向量 X^{k} 中，绝大多数变量是节点电位，少数变量是支路电流。

在 SPICE 程序中应用了既判电压又判电流的收敛标准。它先判非线性器件支路的支路电压和支路电流是否满足收敛判据，若不满足则继续迭代；若满足了，再判全部解向量 X 前后两次的差是否满足收敛条件。SPICE 程序中取 $\varepsilon = 10^{-6}$V 和 10^{-12}A，取 $\varepsilon = 0.001$。当然，收敛判据严格了，所需迭代次数要相应增加，但收敛性则更有保证。

然而，经典的 N-R 方法在求解具体电路时常常会遇到不收敛的情况。N-R 迭代的收敛性在很大程度上取决于：(1) 方程系数矩阵是否具有主对角优势；(2) 初始值 X^{0} 的选取是否合适。可以证明，如果方程系数矩阵具有严格的主对角优势，或者初值 X^{0} 选得足够的好，N-R 方法一定收敛。一般而言，方程系数矩阵的结构取决于电路的形式，很难保证一定具有主对角优势。而且除了一些简单的电路外，要给出足够好的初值是极困难的。N-R 方法不收敛的情况常常表现为两个方面：(1) 在某个迭代点上 $\left|f_{i}(X^{k+1})\right|$ 的值过大，计算机产生数值溢出，我们称之为发散。如图 4.4.1 所示是一个简单二极管电路的 N-R

迭代过程。如果初始值或某个中间迭代值较小,则在得到下一次迭代值 U^{k+1} 时,在 U^{k+1} 处的二极管电流 $f(U^{k+1})$ 太大,以致产生数值溢出。这种情况,在含半导体器件电路的 N-R 迭代中经常会发生。(2) 在迭代过程中可能引起解的振荡。以图 4.4.2 为例,这是一个隧道二极管特性曲线,真正解是 U^*。当某次迭代值取值为 U^k 时,二极管线性化方程用切线 G^k 代替,得到下一次迭代值 U^{k+1},在 U^{k+1} 处切线为 G^{k+1},得到下一次迭代点 $U^{k+2}=U^k$。如此反复,迭代值在 U^k 与 U^{k+1} 处振荡,不能收敛于真正解 U^*。不仅仅是隧道二极管如此,其它一般的晶体管电路,在 N-R 迭代过程中也会发生类似的振荡现象。因此,为了防止 N-R 方法的不收敛情况,提出了若干改进 N-R 方法,以改善算法的收敛性。

图 4.4.1　数值溢出实例

图 4.4.2　解振荡的实例

4.5　改进的 N-R 方法

4.5.1　“横取”N-R 方法

经典 N-R 方法进行迭代时,是利用每次迭代求得的非线性支路电压 $\hat{U}^{k+1}$,直接作为下一次迭代的初值,即 $U^{k+1}=\hat{U}^{k+1}$($\hat{U}^{k+1}$ 是第 $k+1$ 次迭代的结果,U^{k+1} 是第 $k+2$ 次迭代的初值)。这时

$$\hat{I}^{k+1}=I_S(e^{\hat{U}^{k+1}/V_T}-1)$$

习惯上将经典 N-R 方法称为“竖取”N-R 法,如图 4.5.1 中↑方向所示。此法常常由于 I^{k+1} 值太大而产生数值溢出。

下面我们对竖取 N-R 方法进行改进。先在迭代点 $\hat{U}^{k+1}$,沿水平方向求得 $\hat{I}^{k+1}$,如图 4.5.1 中←方向所示,再用非线性支路特性方程的反函数求得 U^{k+1},作为下一次迭代的初值。这种改进的 N-R 方法也称为“横取”N-R 方法。对二极管类元件该算法公式为

$$\hat{I}^{k+1}=I^k+G^k(\hat{U}^{k+1}-U^k)$$

$$U^{k+1}=f^{-1}(\hat{I}^{k+1})=V_T\ln\frac{\hat{I}^{k+1}}{I_S}+1 \qquad (4.5.1)$$

其中 $f(\,)$ 是二极管支路特性函数,即

$$f(U)=I_S(e^{U/V_T}-1) \qquad (4.5.2)$$

实际应用中常交替采用“横取”和“竖取”N-R 方法，其算法描述如下：

$$U^{k+1}=\begin{cases}\hat{U}^{k+1} & \text{当 } \hat{U}^{k+1}<U^{k} \quad (4.5.3)\\ f^{-1}(\hat{I}^{k+1}) & \text{当 } \hat{U}^{k+1}>U^{k} \quad (4.5.4)\end{cases}$$

还可以引入一个可调的经验常数 α 适当组合上述两种方法，得到一个新的校正公式：

$$U^{k+1}=f^{-1}[(1-\alpha)f(U^{k})+(1-\alpha)f'(U^{k})(\hat{U}^{k+1}-U^{k})+\alpha f(\hat{U}^{k+1})] \quad (4.5.5)$$

当 $\alpha=0$ 时，(4.5.5)式即为“横取”N-R 法的算法公式(4.5.4)；当 $\alpha=1$ 时，(4.5.5)式即为经典 N-R 方法的算法公式(4.5.3)；当 α 取 0～1 之间某值时，α 越小越接近于“横取”N-R 法，α 越大越接近经典 N-R 方法。这是一种交替使用电流和电压值做为迭代值的算法，收敛速度较快。

图 4.5.1 “横取”和“竖取”N-R 方法

4.5.2 “四象限”算法

这是继(4.5.3)，(4.5.4)式之后另一种交替使用“竖取”和“横取”N-R 法的算法。我们设定一个电位限定值 V_Q，通过非线性特性曲线上的(V_Q，I_Q)点，把原曲线坐标分为四个象限，如图 4.5.2 所示。“四象限”算法公式如下：

$$U^{k+1}=\begin{cases}f^{-1}(\hat{I}^{k+1}) & (\hat{U}^{k+1},\hat{I}^{k+1}) \text{ 落在第 I 象限}\\ \hat{U}^{k+1} & (\hat{U}^{k+1},\hat{I}^{k+1}) \text{ 落在第 III 象限}\\ V_Q & (\hat{U}^{k+1},\hat{I}^{k+1}) \text{ 落在第 IV 象限}\end{cases} \quad (4.5.6)$$

其中 $\hat{I}^{k+1}$ 仍由前面式(4.5.1)决定。一般迭代结果点($\hat{U}^{k+1}$，$\hat{I}^{k+1}$)不会落在第Ⅱ象限。这种“四象限”算法的主要特点在于第Ⅳ象限的处理。通常当($\hat{U}^{k+1}$，$\hat{I}^{k+1}$)落在第四象限时，无论采用“横取”N-R 法还是“竖取”N-R 法都不容易收敛；采用“竖取”N-R 法可能产生数值溢出；采用“横取”N-R 法可能引起“假收敛”，或收敛速度很慢。这是由于曲线在零点附近斜率较小，斜率变化也很小的缘故。V_Q 是个经验数值，一般取为曲线的最小半径点。在 SPICE 程序中，V_Q 由下式决定：

图 4.5.2 “四象限”算法示意图

$$V_Q=V_T\lg\frac{V_T}{\sqrt{2}I_S} \quad (4.5.7)$$

在实际电路分析中验证，这种“四象限”算法所需迭代次数最少，收敛速度最快。

4.5.3 阻尼算法

经典 N-R 算法的不收敛往往是由于前后两次迭代值之间差过大所致，因此阻尼算法的主导思想就是限制前后两次迭代值之间的差。在阻尼算法中，设阻尼因子 p 为一个本次迭代结果与前一次迭代初值有关的函数，即有

$$U^{k+1} = U^k + p(\hat{U}^{k+1} - U^k) \tag{4.5.8}$$

阻尼因子 p 定义为

$$p = \min\left(1, 0.1 + \frac{C_1}{|\hat{U}^{k+1} - U^k| + C_2}\right) \tag{4.5.9}$$

C_1 和 C_2 为适当选取的常数，是经验数值，一般取 $1 > C_1 > C_2$。当非线性支路第 k+1 次迭代结果 $\hat{U}^{k+1}$ 与前一次迭代初值 U^k 之差较大时，为了避免迭代发散，需加强阻尼作用，则选取较小的 p 值。例如，当 $|\hat{U}^{k+1} - U^k| > 10C_1 - C_2$ 时，$p < 0.2$，因此 U^{k+1} 的变化不会超过 $|\hat{U}^{k+1} - U^k|$ 的 20%。随着 $|\hat{U}^{k+1} - U^k|$ 的减少，百分比将逐渐增加。当 $\hat{U}^{k+1}$ 接近 U^k 时，显然不再需要阻尼 $\hat{U}^{k+1}$ 的变化，故取 $p = 1$。这样由 $|\hat{U}^{k+1} - U^k|$ 值自动地计算阻尼因子 p 值，有利于防止迭代发散或振荡。另外，可针对不同类型电路适当选取 C_1 和 C_2 值，既可调整 U^{k+1} 的变化幅度，又可以改变结束阻尼的时间。

4.5.4 高阶校正法

前面介绍“横取”N-R 方法时曾提出，当支路特性存在反函数时，可用如下公式计算下一次迭代的初值：

$$U^{k+1} = f^{-1}(\hat{I}^{k+1}) \tag{4.5.10}$$

$$\hat{I}^{k+1} = f(U^k) + f'(U^k)(\hat{U}^{k+1} - U^k) \tag{4.5.11}$$

其中，f( )是支路特性函数。

(4.5.10)和(4.5.11)相当于在一定条件下，不采用第 k+1 次迭代结果 $\hat{U}^{k+1}$ 为第 k+2 次迭代的初值，而是利用函数 f(U)对 $\hat{U}^{k+1}$ 校正一次，得到校正值 U^{k+1} 作为下一次迭代的初值。校正公式是 f(U)在 U^k 处的一阶台劳级数。因此(4.5.11)式是一阶校正公式。

还可以采用高阶校正公式，例如 n 阶校正公式

$$\hat{I}^{k+1} = f(U^k) + f'(U^k)(\hat{U}^{k+1} - U^k) + \frac{f''(U^k)}{2!}(\hat{U}^{k+1} - U^k)^2 + \cdots + \frac{f^{(n)}(U^k)}{n!}(\hat{U}^{k+1} - U^k)^n \tag{4.5.12}$$

通常取二阶即可，此时有

$$U^{k+1} = f^{-1}\left[f(U^k) + f'(U^k)(\hat{U}^{k+1} - U^k) + \frac{1}{2}f''(U^k)(\hat{U}^{k+1} - U^k)^2\right] \tag{4.5.13}$$

我们以二极管为例,推导二阶校正公式。二极管支路特性关系为

$$I_D = f(U_D) = I_S(e^{U_D/V_T} - 1)$$

支路特性函数的反函数和一阶、二阶导数如下:

$$f^{-1}(I_D) = V_T \ln I_D/I_S + 1$$

$$f'(U_D) = \frac{I_S}{V_T} e^{U_D/V_T}$$

$$f''(U_D) = \frac{I_S}{V_T^2} e^{U_D/V_T}$$

代入(4.5.13)式,即有

$$U_D^{k+1} = U_D^k + V_T \ln\left(1 + \frac{\hat{U}_D^{k+1} - U_D^k}{V_T}\right)\left(1 + \frac{\hat{U}_D^{k+1} - U_D^k}{2V_T}\right) \qquad (4.5.14)$$

求得 $k+1$ 次迭代值 $\hat{U}_D^{k+1}$ 后,可利用(4.5.14)式得到其二阶校正值 U_D^{k+1},作为第 $k+2$ 次迭代初值。使用这种二阶校正法,当 U^{k+1} 接近真值 U^* 点附近时非常有效,能加速收敛。上述一维高阶校正法的思想,可以推广到多维情况。

4.6 其它改善收敛性的算法

我们将经典的 N-R 方法与各种改进 N-R 方法相结合,去求解非线性电路的直流工作点时,一般来说,其收敛性是可以得到保证的。但在有些情况下,仍会发生解不收敛的现象。这种不收敛,可能是由于有电路连接错误,或元件值及模型参数有错造成的,此外,某些双稳态电路、无稳态电路、控制开关电路、具有强正反馈的电路和某些运放电路也会在直流分析时遇到不收敛的问题。通常可以用以下几种方法克服和改善这类不收敛问题。

(1) 采用源连续变化的算法(称为源步进法);

(2) 采用伪瞬态分析的方法(称为伪瞬态法);

(3) 采用预先设置电路节点电压的方法(称为初始条件松弛法);

(4) 采用将电路中某些非线性器件截止的方法(称为状态松弛法);

(5) 采用限制非线性支路最小电导值的方法(称为 G_{min} 步进法)。

这些方法仍然要以 N-R 方法或各种改进 N-R 方法为基础。下面我们简要介绍这几种方法。

1. 源步进法

一般方程组的解依赖于源,源连续变化,解也相应连续变化,若源变化剧烈,就可能产生解不收敛的情况。源步进法的主导思想就是让源逐渐变化,以保证 N-R 迭代的收敛。

设电路中实际源(独立电压源或电流源)为 S,进行直流分析时取源为

$$V_E = \alpha S \qquad (4.6.1)$$

让 α 逐步从 0 变到 1。当 $\alpha = 0$ 时,显然方程组的解为 0。在以后的每一步,逐步增加 α 但要使 N-R 法能收敛。每步求得的收敛解,作为下一次求解的初值。只要 a 选取得适当,就可以既得到收敛解,又可能保证总的计算时间不至于过长。

源步进法的算法步骤如下:

(1) 令 α_{min} = 1/64, 给定初值 α, 例如: α = 1/8, 系数 F = 1/8, α = α* F。

(2) 如 $\alpha \geq 1/64$, 进行 N-R 迭代。如收敛, F = 2, 转(3); 如不收敛, F = 2* F, α = α* F 继续迭代。

如 $\alpha < 1/64$, 转(6)。

(3) α = F* α, 如 $\alpha < 1$, 进行 N-R 迭代, 转(4); 如 $\alpha \geq 1$, 则令 α = 1, 进行 N-R 迭代, 转(4)。

(4) 如果收敛, 且 α = 1, 则得收敛解, 返回。如果收敛, 但 $\alpha < 1$, 转(3)。

如不收敛, 转(5)。

(5) F = $\overline{F}$, 如 F < 1.0001, 转(6); 否则 α = αF, 进行 N-R 迭代。若收敛, 转(3); 若不收敛, 转(5)。

(6) 源步进法失败, 返回。

源步进法对于振荡电路, 以及多稳态电路的收敛有帮助。但采用源步进法后, 所需迭代次数显著增加, 计算时间也会增长。在 SPICE 程序中, 当直流迭代不收敛时, 自动转入用源步进法迭代, 以帮助收敛。

2. 伪瞬态法

伪瞬态法是在直流分析时, 在电路中引入伪电容和伪电感, 然后进行瞬态分析, 将电路达到稳态时的解, 作为直流工作点的解。

具体步骤是: 在每个独立电压源和每个非线性的电压相关支路上串联一个伪电感, 例如取为 1H; 在每个独立电流源和每个非线性电流相关支路上并联一个伪电容, 如取为 1F。这些伪元件上的初始条件取为零 。然后, 采用向后欧拉积分公式, 对这个伪电路进行瞬态分析。在分析过程中, 不考虑解的变化过程, 也不管其截断误差的大小, 积分时间步长的选择也不受精度要求的限制, 只要 N-R 方法收敛, 就可以尽可能地选择较大的时间步长。例如, 当某个时间点上, N-R 迭代次数超过 10 次还没收敛, 则积分步长缩减到原步长的 1/8; 若迭代次数小于 5 次就收敛了, 则积分步长放大一倍。这样求得的稳态解即为直流解。

伪瞬态分析对于紧耦合型电路的收敛较为有效。

3. 初始条件松弛法

在求直流工作点前, 先预先设定电路中某些节点(或全部节点)的初始电位值。在进行直流分析过程中, 在这些节点上连接等于初始电位值的电压源, 并进行 N-R 迭代, 直至收敛。然后将这些值的强度松弛, 以致能由计算出的节点电位值取代之, 继续进行 N-R 迭代, 直至达到新的直流工作点。因此, 最初设置的节点电位值并不影响最终的直流工作点, 而仅仅起到帮助收敛的作用。

在 SPICE 程序中, 是通过节点电压设置卡(.NODESET)来实现这一算法的。

4. 状态松弛法

最初, 将电路中某些(或全部)二极管和三极管都设定为截止(OFF)状态, 即将其端电压设置为零, 在这种状况下进行直流分析。直流收敛以后, 逐次地撤掉 OFF 状态, 继续进行 N-R 迭代, 以得到这些器件端电压的正确值。因此, 最初设定非线性元件为截止状态,

仅仅是一种初始条件,并不影响最终的直流工作点。

SPICE 程序中采用在非线性器件卡上设置 OFF 可选项,来实现这一状态松弛算法,这种算法对大规模 MOS 和 BIMOS 电路特别有效,也有助于含有超过上千个晶体管电路的模拟。

5. G_{min} 步进法

G_{min} 参数是 SPICE 程序中非线性支路的一个给定最小电导值,程序中设其隐含值为 10^{-12}S。它通常并联在非线性支路上。当采用 N-R 迭代时,在每一步迭代中,都将非线性支路特性转化为等效的线性电导和电流源。一般情况下,等效线性电导大于 G_{min},故 G_{min} 不起作用。例如图 4.6.1 所示二极管的线性化模型。

G_{min} 步进算法步骤如下。先设置 G_{min} 为一个较大的值,例如 1S,则在 N-R 算法将非线性特性转化为线性模型过程中,程序将所有非线性支路都用一值为 1S 的线性电导取代,计算出这个线性电路的直流解。然后,G_{min} 逐步减小,当 G_{min} 减小到小于非线性支路的等效电导 G_{DM}^{k} 时,G_{min} 不再起作用,电路开始进行正常的 N-R 迭代,并由此求解出新的直流解。这种方法使电路从线性电路起步,然后逐渐返回到原始非线性电路的直流解。从而改善了非线性迭代的收敛性。在 SPICE 程序中,可以在可选项语句(.OPTIONS)中设置 G_{min} 值。

图 4.6.1　二极管 N-R 线性化模型

4.7　直流非线性分析流程和分析实例

直流非线性分析流程是在 4.1 节介绍的直流线性分析流程的基础上增加了下面两方面内容:

(1) 采用 N-R 方法和各种改进 N-R 方法,将非线性器件的非线性方程简化为每个迭代点上的线性方程,也就是说将非线性模型线性化。

(2) 进行 N-R 算法的迭代循环,直到满足迭代误差要求,即收敛为止。

非线性直流分析流程如图 4.7.1 所示。图中的 k_{max} 是直流迭代次数的限制,在 SPICE 程序中由可选项语句(.OPTIONS)中的 ITL1 变量确定,其隐含值是 100,可以通过重新设定 ITL1 值来确定。

例 4.7.1　计算图 4.7.2 所示差分对电路的直流输入、输出电阻以及直流传输特性。其输入数据文件如下:

```
BJT   Differential   Amplifier
Q1   4   3   5   MOD    ;图中的 T1
Q2   6   7   8   MOD    ;图中的 T2
RC1   4   9   3K
RC2   6   10   3K
VC1   1   9   0          ;图中的 EC1
```

图 4.7.1　直流分析流程图

图 4.7.2　差分对电路

图 4.7.3　传输特性曲线

```
VC2   1   10   0        ;图中的 EC2
RE1   5   11   RMOD 1
RE2   8   11   ROMD 1
```

```
RE   11   2   1K
VCC   1   0   12
VDD   2   0   - 6
VI1   3   0   0        ;图中的 UI1
VI2   7   0   0        ;图中的 UI2
.MODEL   MOD   NPN   IS= 1E- 15   BF= 80   TF= 10N
.MODEL   RMODRES(R= 5)
.OP
.TF   V(6)   VI1
.DC   VI1   - 0.5   0.5   0.01   RES   RMOD(R)   LIST   5   50
.PROBE
.END
```

计算出的直流工作点如下:

NODE	VOLTAGE	NODE	VOLTAGE	NODE	VOLTAGE	NODE	VOLTAGE
(1)	12. 0000	(2)	- 6. 0000	(3)	0. 0000	(4)	4. 2258
(5)	- . 7393	(6)	4. 2258	(7)	0. 0000	(8)	- . 7393
(9)	12. 0000	(10)	12. 0000	(11)	- . 7524		

差分对管的电压、电流和等效参数如下:

* * * * BIPOLAR JUNCTION TRANSISTORS

NAME	Q1	Q2
MODEL	MOD	MOD
IB	3. 24E- 05	3. 24E- 05
IC	2. 59E- 03	2. 59E- 03
VBE	7. 39E- 01	7. 39E- 01
VBC	- 4. 23E+ 00	- 4. 23E+ 00
VCE	4. 97E+ 00	4. 97E+ 00
BETADC	8. 00E+ 01	8. 00E+ 01
GM	1. 00E- 01	1. 00E- 01
RPI	7. 98E+ 02	7. 98E+ 02
RX	0. 00E+ 00	0. 00E+ 00
RO	1. 00E+ 12	1. 00E+ 12
CBE	1. 00E- 09	1. 00E- 09
CBC	0. 00E+ 00	0. 00E+ 00
CBX	0. 00E+ 00	0. 00E+ 00
CJS	0. 00E+ 00	0. 00E+ 00
BETAAC	8. 00E+ 01	8. 00E+ 01
FT	1. 59E+ 07	1. 59E+ 07

电路的直流增益和输入、输出电阻如下:

```
V(6)/VI1 =   9.898E+ 01
INPUT   RESISTANCE   AT   VI1 =   2.389E+ 03
OUTPUT   RESISTANCE   AT   V(6) =   3.000E+ 03
```

发射极电阻分别为 5Ω和 50Ω时,电路中差分对管集电极电流随输入电压变化曲线(直流传输曲线)如图 4.7.3 所示。图中①,① 分别是 R_{E1}为 5Ω,50Ω时的 I_{C1}曲线;②,②′分别是 R_{E2}为 5Ω,50Ω时的 I_{C2}曲线。可以看出,当发射极电阻增大时,输出动态范围加大。

习　　题

4.1　推导双极型晶体管注入型 EM1 模型的直流伴随模型,写出相应的数学公式,画出伴随模型图,并将其对节点方程组的贡献以送值表的形式写出来。

4.2　题图 4.2 所示简单二极管电路中二极管参数为:$I_S = 10^{-10}$A,$V_T = 0.025$V,$R_S = 0$;电流源 I = 0.66A,电阻 R = 0.4Ω。请列出二极管的直流伴随模型和电路的节点方程组,并用 N-R 方法求解此电路的直流工作点,设初值 $U_D^0 = 0.20$V,请写出迭代次数 k = 0, 1, 2 时的求解过程。

题图　4.2

4.3　比较经典 N-R 方法,“横取”N-R 方法和“四象限”法的优劣。

4.4　请画出采用“四象限”算法的直流非线性迭代的程序流程图。

4.5　列出题图 4.5 所示电路第 k 次迭代的混合方程组。BJT 晶体管采用混合 π型 EM1 直流伴随模型。

题图　4.5

4.6　在直流非线性迭代中,二极管的特性方程为 $I_D = I_S\left(e^{U_D/V_T} - 1\right)$,其中 $I_S =$

10^{-10}A, V_T = 0.026V。二极管直流伴随模型中, R_s = 0。

(1) 若第 k 次迭代初值 U_D^k = 0.3V, 计算第 k 次迭代的二极管直流伴随模型参数, 并在题图 4.6 所示的迭代示意图上标出模型参数。

题图 4.6　迭代示意图

(2) 若第 k 次迭代结果 $\hat{U}_D^{k+1}$ = 0.5V, 请用“横取”N-R 方法计算第 k+1 次迭代的模型参数。

4.7　用 PSpice 程序计算题图 4.7 所示电路的直流工作点。应用电路知识(或灵敏度分析)调整电路参数, 使输出中点(A 点)电位为 6.00±0.05V。

题图　4.7

4.8　用 PSpice 程序计算题图 4.8 所示电路的直流工作点。

(1) 调节哪个电阻可以调节 A 点的电位?

(2) 调节哪个电阻可以调节输出中点 B 的电位? 请将 B 点电位调到正确位置。

4.9　用 PSpice 程序计算题图 4.9 所示差分对电路的如下特性:

题图 4.8

题图 4.9

(1) 直流传输特性;

(2) 差模电压增益;

(3) 差模输入和输出电阻;

(4) 共模输入电阻。

4.10 书后附录中是一个用 C 语言写的直流分析的教学程序, 读懂该程序, 并添加如下模型:

(1) 电容、电感的直流模型;

(2) 受控源 CCCS、VCVS 和 CCVS 的改进节点法直流线性模型。

4.11 在附录的直流分析教学程序中, 二极管采用的是“四象限”算法。请在二极管函数中增加如下算法:

(1) 经典 N-R 方法;

(2) 阻尼法;

(3) 高阶校正法。

通过具体实例比较这三种方法与“四象限”算法在收敛速度上的优劣。

4.12 在附录的直流分析教学程序中增添双极型晶体管函数, 具体要求是:

(1) 双极型晶体管采用简化的 EM2 模型, 即不考虑 $R_{CC'}$ 和 $R_{EE'}$;

(2) 输入信息中应包括必需的晶体管模型参数和 U_{BE}、U_{CE} 的初始值;

(3) 采用“四象限”算法;

(4) 能同时处理 NPN 和 PNP 型 BJT 晶体管。

第5章　瞬 态 分 析

5.1　引　　言

电路的瞬态分析就是求电路的时域响应。它可以是在给定激励信号情况下求电路输出的时间响应、延迟特性，也可能是在没有任何激励信号的情况下求振荡电路的振荡波形、振荡周期。瞬态分析时，电路是由一组常微分方程来描述，因此瞬态分析的实质就是如何求解常微分方程。对于线性电路，可采用拉氏变换的方法，通过网络的传输函数求得输出时域响应。但拉氏变换的方法只适用于简单的线性电路，当电路比较复杂时，寻求传输函数也十分困难，更何况大多数实际电路都是非线性电路。因此，在电路 CAD 中，瞬态分析是采用数值方法求解常微分方程，它既适用于线性电路，也适用于非线性电路。求解微分方程的数值方法也称为数值积分方法。用这种方法解决电路问题包括三部分内容：器件模型、电路方程的建立和方程的求解。前两部分内容我们已在前面章节中作了介绍，在这一章中我们着重讨论求解常微分方程的各种数值积分方法以及相应的储能元件的模型。

瞬态分析时，电路方程可以表示为如下一阶常微分方程：

$$x' = f(x,t) \qquad t_0 \leq t \leq T \tag{5.1.1}$$

其初始条件

$$x(t_0) = x_0$$

或写成隐式形式：

$$\begin{aligned} &F(x,x',t) = 0 \qquad t_0 \leq t \leq T \\ &x(t_0) = x_0 \end{aligned} \tag{5.1.2}$$

其中 x 为电路未知变量向量，x' 为 x 对 t 的一阶导数向量，x_0 是初值向量，t 是定义在时间区间 $[t_0, T]$ 上的时间变量。我们要求未知向量 x 在 $[t_0, T]$ 时间域内的数值解。

我们以两个简单的实例来说明瞬态分析的求解过程。

例 5.1.1　一个简单的线性电路及输入信号波形如图 5.1.1 所示。在 $t=0$ 时，信号源 $E(t)$ 由零突跳至 E，求在 $[t_0, T]$ 时间区间内，电容两端电压 $U_C(t)$ 的瞬态响应。其中，$R = 1\text{k}\Omega$，$C = 10\mu\text{F}$，$t_0 = 0$，$T = 0.1\text{s}$，$U_C(t_0) = 0$。

图 5.1.1　简单线性电路及输入信号波形

(1) 列代数微分方程

$$\begin{aligned} IR + U_C &= E \\ C\frac{dU_C}{dt} &= I \end{aligned} \tag{5.1.3}$$

其初始条件 $U_{C,0}=0$

(2) 将时间 t 离散化。把$[t_0, T]$分为 $t_0, t_1, t_2, \cdots, t_n, \cdots, T$ 等离散点，相邻两点之间的时间间隔 $h_n = t_{n+1} - t_n$，称为步长。这里取 $h = 1\text{ms}$。

(3) 在各时间点上将微分方程化为差分方程。这可通过应用各种数值积分方法来实现。假定采用向前欧拉法，即用一阶向前差商来代替方程(5.1.3)中的导数，有

$$\left.\frac{dU_C}{dt}\right|_{t_n} \approx \frac{U_{C,n+1} - U_{C,n}}{h_n}$$

其中 $U_{C,n+1}$和 $U_{C,n}$分别是 t_{n+1}和 t_n 时刻点的近似解，于是(5.1.3)式可化为如下差分方程组：

$$\begin{aligned} I_n R + U_{C,n} &= E \\ \frac{C}{h}(U_{C,n+1} - U_{C,n}) &= I_n \\ U_{C,0} &= 0 \end{aligned}$$

化简后得

$$\begin{aligned} U_{C,n+1} &= \frac{hE}{RC} + \left(1 - \frac{h}{RC}\right) U_{C,n} \\ U_{C,0} &= 0 \end{aligned} \tag{5.1.4}$$

(4) 解方程：将 R, C, h 等值代入方程(5.1.4)，由 $t_0 = 0$ 开始计算，则(5.1.4)式可表示为

$$\begin{aligned} U_{C,n+1} &= 0.1E + 0.9U_{C,n} \\ U_{C,0} &= 0 \end{aligned}$$

$t_0 = 0$	$U_{C,0} = 0$
$t_1 = h = 1\text{ms}$	$U_{C,1} = 0.1E + 0.9U_{C,0} = 0.1E$
$t_2 = 2\text{ms}$	$U_{C,2} = 0.1E + 0.9U_{C,1} = 0.19E$
$t_3 = 3\text{ms}$	$U_{C,3} = 0.1E + 0.9U_{C,2} = 0.271E$

一直计算到 $t = T$ 为止。

由于这是一个简单的电路，我们可以用解析方法求解。方程(5.1.3)的解析解为 $U_C(t) = E\left(1 - e^{-\frac{t}{RC}}\right)$。图 5.1.2 中绘出了方程的解析解与数值解。如果认为解析解是精确解，那么可以看出数值解是存在一定误差的。

例 5.1.2　图 5.1.3 表示一个含有非线性二极管的简单电路。求二极管两端电压 $U_D(t)$ 在时间$[t_0, T]$内的瞬态响应。

图 5.1.2　例 5.1.1 电路中 $U_C(t)$ 的解　　图 5.1.3　简单二极管电路

设二极管模型等效成一个非线性电流源 I_D 与一个电容 C_D 并联，忽略其体电阻，则电路可等效为图 5.1.4 所示形式。

(1) 列微分方程

$$\frac{U_D}{R} + C_D \frac{dU_D}{dt} + I_S \left(e^{U_D/V_T} - 1 \right) - \frac{E_2}{R} = 0 \qquad (5.1.5)$$

其初始条件

$$U_D(0) = E_1$$

(2) 化为差分方程。假设采用向后欧拉公式，即 $\left.\frac{dU_D}{dt}\right|_{t_{n+1}} \approx \frac{U_{D,n+1} - U_{D,n}}{h_n}$，并代入方程

图 5.1.4　图 5.1.3 电路的等效电路　　图 5.1.5　线性等效电路

(5.1.5)，得

$$\frac{U_{D,n+1}}{R} + \frac{C}{h} U_{D,n+1} - \frac{C}{h} U_{D,n} + I_S \left(e^{U_{D,n+1}/V_T} - 1 \right) - \frac{E_2}{R} = 0 \qquad (5.1.6)$$

初始条件

$$U_{D,0} = E_1$$

显见，这是一个非线性代数方程。

(3) 求解非线性方程。由于(5.1.6)式是个非线性方程组，因此在每个时间点上要先将它化为线性代数方程，即用 N-R 方法将非线性电流源 I_D 等效为每个迭代点上的电导 G_{DM}^k 和电流源 I_{DM}^k 的并联，如图 5.1.5 所示。经过若干次迭代，收敛以后的解才能作为当前时间点的瞬态解。

由上面两个实例可以看出，用差分方程逼近微分方程是数值法求解微分方程的关键。可以应用各种数值积分方法将微分方程转化为差分方程。不同的方法对计算的精度、速度以及计算机内存容量的需求等有很大差别。我们在后面几节中将着重介绍各种不同的数

值积分方法,并比较它们在精度、稳定性等方面的优劣。另外,电路方程一般是非线性方程,所以非线性瞬态分析的过程可以理解为:(1)在各时刻点上采用数值积分方法将微分方程化为差分方程,即通过递推过程求得下一时刻的未知量;(2)在每一时刻点用 N-R 方法迭代求解非线性方程。

5.2 常用的数值积分方法

本节介绍几种在实用程序中常用的低阶数值积分方法,它们是向前欧拉法(forward Euler algorithm)、向后欧拉法(backward Euler algorithm)和梯形法(trapezoidal algorithm)。

5.2.1 向前欧拉法

1. 计算公式

设 $x(t)$ 是微分方程

$$x' = f(x,t)$$
$$x(t_0) = x_0$$

的精确解,将 $x(t)$ 在 t_n 处展成台劳级数,即有

$$x(t) = x(t_n) + x'(t_n)(t-t_n) + \frac{1}{2!}x''(t_n)(t-t_n)^2 + \cdots$$
$$+ \frac{1}{k!}(t-t_n)^k \frac{d^k}{dt^k}x(t)\Big|_{t=\xi} \qquad t_n < \xi < t_{n+1}$$

当 $t = t_{n+1}$ 时,取 $k=2$,则精确解为

$$x(t_{n+1}) = x(t_n) + x'(t_n)h_n + \frac{1}{2!}h^2x''(\xi) \qquad t_n < \xi < t_{n+1} \tag{5.2.1}$$

其中,

$$h_n = t_{n+1} - t_n$$

假定 $x_n = x(t_n)$, $x_n' = x'(t_n)$ 是精确解,并取(5.2.1)式的前两项作为 $x(t_{n+1})$ 的近似值 x_{n+1},即得

$$x_{n+1} = x_n + h_n x_n' \tag{5.2.2}$$

这就是向前欧拉公式,它是一阶的显式数值积分公式。

2. 局部截断误差

由于在 t_{n+1} 时刻点用差分方程(5.2.2)代替微分方程(5.2.1),而解得 $x(t_{n+1})$ 的近似值 x_{n+1},故精确解与近似解之间的差 $x(t_{n+1}) - x_{n+1}$ 通常称为局部截断误差(local truncation error),记作 E_T。局部截断误差的大小标志了数值积分公式的精度。

向前欧拉公式的局部截断误差为

$$E_T = x(t_{n+1}) - x_{n+1} = \frac{1}{2}h_n^2x''(\xi) \tag{5.2.3}$$

即向前欧拉法的局部截断误差是二阶的。

我们在考虑 t_{n+1} 时刻的局部截断误差时,是假定在前面 $t_n, t_{n-1}, \cdots$ 时刻点上的解 $x_n, x_{n-1}, \cdots$ 是精确解。实际上,在解差分方程时,每一时刻所计算出的解都是近似解,都存在

局部截断误差。另外。由于计算机的字长有限,每个解同时也存在有舍入误差。所以,数值积分在 t_{n+1} 时刻求得 $x(t_{n+1})$ 的近似解 x_{n+1},它与精确解的误差由两部分组成:(1)在 t_{n+1} 时刻用差分方程代替微分方程所产生的局部截断误差 $E_{T,n+1}$。(2)由于 t_{n+1} 以前时刻解的局部截断误差和舍入误差等的影响而产生的累积舍入误差。

3. 稳定性

当前面若干时刻解的误差对后面时刻点上解的影响不随积分步数的增加而增加,即累积舍入误差是有界的,则称这种数值积分方法是稳定的,否则称这种积分方法是不稳定的。

通常用微分方程

$$x' = \lambda x \tag{5.2.4}$$

作为试验方程来判断各种数值积分方法的稳定性。方程(5.2.4)的精确解为

$$x(t) = x_0 e^{\lambda t} \tag{5.2.5}$$

其中 $x_0 = x(0)$,即 $t = 0$ 时的初值;λ 是复数,为微分方程的特征根。当 $\mathrm{Re}\lambda < 0$ 时,这个方程所代表的系统本身是稳定的,这时讨论数值积分方法是否稳定才有意义。因此,我们在假定 $\mathrm{Re}\lambda < 0$ 的条件下讨论数值积分方法的稳定性。

将试验方程(5.2.4)应用于向前欧拉公式,有

$$x_{n+1} = x_n + hx'_n = x_n + h(\lambda x_n) = (1 + h\lambda)x_n \tag{5.2.6}$$

若 x_n 存在有误差 ε,即 $x_n = x(t_n) - \varepsilon$,代入(5.2.6)式,得

$$\begin{aligned} x_{n+1} &= (1 + h\lambda)[x(t_n) - \varepsilon] \\ &= (1 + h\lambda)x(t_n) - (1 + h\lambda)\varepsilon \\ &= (1 + h\lambda)x(t_n) - \varepsilon_{n+1} \end{aligned}$$

其中 ε_{n+1} 表示 t_{n+1} 时刻的解 x_{n+1} 的误差。若希望 ε_{n+1} 小于 ε,即 $|(1+h\lambda)\varepsilon| < |\varepsilon|$,则需

$$|1 + h\lambda| < 1$$

即

$$h < \frac{2}{|\lambda|} \tag{5.2.7}$$

这就是向前欧拉法的稳定条件。

如果试验方程是一组微分方程,则可推导出

$$h < \frac{2}{|\lambda_{max}|}$$

其中 λ_{max} 是方程组中最大的 λ。

图 5.2.1 中绘出了在 $h\lambda$ 复平面上向前欧拉法的稳定区域,即图中的阴影部分,可以看出其稳定域较小。

由稳定性的要求,步长 h 必须小于允许值,否则误差会逐步增大,导致计算结果错误。因此应用向前欧拉法时,选择步长时除了要满足局部截断误差小于误差限以外,还必须考虑满足稳定性要求。

图 5.2.1 向前欧拉法稳定域

向前欧拉法的稳定域小,所以除了在起步时应用

它以外,其它情况下很少应用。

5.2.2 向后欧拉法

1. 计算公式

设 $x(t)$ 是微分方程

$$x' = f(x,t)$$
$$x(t_0) = x_0$$

的精确解。将 $x(t)$ 在 t_{n+1} 点展成台劳级数,得

$$x(t) = x(t_{n+1}) + x'(t_{n+1})(t - t_{n+1}) + \frac{1}{2!}x''(t_{n+1})(t - t_{n+1})^2$$
$$+ \cdots + \frac{1}{k!}(t - t_{n+1})^k \frac{d^k}{dt^k}x(t)\Big|_{t=\xi} \qquad t = \xi < t_{n+1}$$

当 $t = t_n$ 时,取 $k = 2$,则精确解为

$$x(t_n) = x(t_{n+1}) - x'(t_{n+1})h_n + \frac{1}{2}h_n^2 x''(\xi) \qquad t_n < \xi < t_{n+1}$$

或写为

$$x(t_{n+1}) = x(t_n) + h_n x'(t_{n+1}) - \frac{1}{2}h_n^2 x''(\xi) \tag{5.2.8}$$

取前两项作为 $x(t_{n+1})$ 的近似值 x_{n+1},即

$$x_{n+1} = x_n + h_n x'_{n+1} \tag{5.2.9}$$

这就是向后欧拉公式。它是一个一阶隐式积分公式。

2. 局部截断误差

向后欧拉法的局部截断误差为

$$E_T = x(t_{n+1}) - x_{n+1} = -\frac{1}{2}h_n^2 x''(\xi) \tag{5.2.10}$$

即向后欧拉公式的局部截断误差是二阶的,与向前欧拉法的相同。

3. 稳定性

将试验公式 $x'_{n+1} = \lambda x_{n+1}$ 代入向后欧拉公式(5.2.9),有

$$x_{n+1} = x_n + h_n x'_{n+1}$$
$$= x_n + h_n(\lambda x_{n+1})$$

即

$$x_{n+1} = \frac{x_n}{1 - h\lambda} \tag{5.2.11}$$

若 x_n 的误差为 ε,即 $x_n = x(t_n) - \varepsilon$,代入(5.2.11)式,得

$$x_{n+1} = \frac{x(t_n) - \varepsilon}{1 - h\lambda} = \frac{x(t_n)}{1 - h\lambda} - \varepsilon_{n+1}$$

若希望 $|\varepsilon_{n+1}|$ 小于 $|\varepsilon|$,则需

$$\frac{1}{|1 - h\lambda|} < 1 \tag{5.2.12}$$

这就是向后欧拉法的稳定条件。由稳定条件得到的向后欧拉法的稳定域如图 5.2.2 阴影

区所示。因为已经假定 $\mathrm{Re}\lambda < 0$, h 又必须是正实数,所以 h 不论取任何实数都能满足(5.2.12)式要求的稳定条件。

图 5.2.2　向后欧拉法的稳定域　　图 5.2.3　向前和向后欧拉公式的几何意义

在应用向后欧拉公式时,步长的选择不必考虑稳定性的要求。当然 h 也不能过大,因为还要满足截断误差小于误差限的要求。

4. 向前欧拉法与向后欧拉法的几何意义

向前欧拉法与向后欧拉法同属于一阶数值积分方法,它们的局部截断误差都是二阶的,但后者的稳定性大大优于前者。

向前欧拉和向后欧拉公式可以用一维图形直观地表示出它们的几何性质,如图 5.2.3所示。图中 $\hat{x}_{n+1} = x_n + h_n x_n'$ 是用向前欧拉公式计算的,它利用了前一个时间点 t_n 上 $x(t)$ 的切线。$\check{x}_{n+1} = x_n + h_n x_{n+1}'$ 是用向后欧拉公式计算的,它利用了后一个时间点 t_{n+1} 上 $x(t)$ 的切线。

5.2.3　梯形法

1. 计算公式与局部截断误差

梯形积分公式可以用推导向后欧拉公式同样的方法,由台劳级数展开式导出。把 x_{n+1} 和 x'_{n+1} 展或台劳级数,有

$$x_{n+1} = x_n + h_n x_n' + \frac{1}{2}h_n^2 x_n'' + \frac{1}{6}h_n^3 \frac{d^3}{dt^3}x(\xi) \tag{5.2.13}$$

$$x_{n+1}' = x_n' + h_n x_n'' + \frac{1}{2}h_n^2 \frac{d^3}{dt^3}x(\xi) \tag{5.2.14}$$

从(5.2.14)式中计算出 x_n'',代入(5.2.13)式,得

$$x_{n+1} = x_n + \frac{h_n}{2}(x_{n+1}' + x_n') - \frac{h_n^3}{12}x(\xi) \qquad t_n \leq \xi < t_{n+1}$$

则梯形公式为

$$x_{n+1} = x_n + \frac{h_n}{2}(x_{n+1}' + x_n') \tag{5.2.15}$$

梯形公式的局部截断误差为三阶,即

$$E_T = -\frac{1}{12}h_n^3 x'''(\xi) \qquad (5.2.16)$$

实际上,梯形公式是向前欧拉公式和向后欧拉公式的算术平均式。通过平均化方法,消除了误差的主要部分$\pm\frac{h_n^2}{2}x''(\xi)$,而获得了更高的精度。

2. 稳定性

设试验方程为

$$x'_{n+1} = \lambda x_{n+1}$$

$$x'_n = \lambda x_n$$

将试验方程代入梯形公式(5.2.15),有

$$x_{n+1} = \frac{1+\frac{h\lambda}{2}}{1-\frac{h\lambda}{2}}x_n \qquad (5.2.17)$$

假定x_n的误差为ε,即$x_n = x(t_n) - \varepsilon$,代入(5.2.17)式,得

$$x_{n+1} = \frac{1+\frac{h\lambda}{2}}{1-\frac{h\lambda}{2}}[x(t_n) - \varepsilon]$$

则有

$$\varepsilon_{+1} = \frac{1+\frac{h\lambda}{2}}{1-\frac{h\lambda}{2}}\varepsilon$$

若希望$|\varepsilon_{+1}|$小于$|\varepsilon|$,需有

$$\left|\frac{1+\frac{h\lambda}{2}}{1-\frac{h\lambda}{2}}\right| < 1 \qquad (5.2.18)$$

这就是梯形法的稳定条件。由于$\mathrm{Re}\lambda < 0$,且$h > 0$,所以h不论取何值都能满足(5.2.18)的稳定性条件。梯形法的稳定域如图5.2.4所示阴影区域。

当采用固定步长时,(5.2.17)式可以写为以下形式:

$$x_{n+1} = \left(\frac{1+\frac{h\lambda}{2}}{1-\frac{h\lambda}{2}}\right)^n x_0 \qquad (5.2.19)$$

当$|h\lambda| > 2$时,此式中的$1+\frac{h\lambda}{2}$为负值。此时,当n是偶数时,x_{n+1}为正值;当n为奇数时,x_{n+1}为负值。也就是说,方程的近似解x_{n+1}将振荡地趋于精确解。一般认为,梯形法的这种振荡特性对求解过程没有害处。

图 5.2.4 梯形法稳定域

梯形公式属于隐式二阶公式,其局部截断误差是三阶的,比向前和向后欧拉法都精

确,其稳定性也较好。

5.2.4 多步法

前面所介绍的数值积分公式都是采用递推法,即用前一个或几个时刻点的数值计算当前时刻点的值。如果求 x_{n+1} 时,只要知道前一时刻点的值 x_n 就够了,这种方法称为单步法。如果求 x_{n+1} 时,需已知前面若干时刻点的值 x_n, x_{n-1}、x_{n-2}, …,则称这种方法为多步法。多步法的数值积分公式可用多项式逼近的方法推导出来,它的一般形式为

$$x_{n+1} = \sum_{i=1}^{p} a_i x_{n-i+1} + h_n \sum_{i=0}^{p} b_i x'_{n-i+1} \tag{5.2.20}$$

其中 p 为积分公式的阶,也是积分的步数。$p=1$ 即为单步法,如向前和向后欧拉法都属于单步法。$p>1$ 为多步法。若 $b_0=0$,(5.2.20)为显式公式,即右端项中不含有未知的 x'_{n+1} 项。若 $b_0 \neq 0$,(5.2.20)为隐式公式,即公式中含有 x'_{n+1} 项,需求解非线性方程组才能求得解 x_{n+1}。

向前欧拉法是单步法,其公式是一阶显式公式。在(5.2.20)式中,当 $p=1$, $a_1=1$, $b_1=1$,其它系数均为零时,即为向前欧拉公式 $x_{n+1}=x_n+h_n x'_n$。向后欧拉法也属于单步法,其公式是一阶隐式积分公式。在式(5.2.20)中,当 $p=1$, $a_1=1$, $b_0=1$,其它系数均为零时,即为向后欧拉公式。梯形公式是当(5.2.20)中, $p=1$, $a_1=1$, $b_0=\frac{1}{2}$, $b_1=\frac{1}{2}$,其它系数为零时所构成的公式。但通常我们称梯形公式为二阶隐式公式。

Gear 法(见 5.5 节)的二阶以上公式都属于多步法的公式,例如三阶基尔公式:

$$x_{n+1} = \frac{18}{11}x_n - \frac{9}{11}x_{n-1} + \frac{2}{11}x_{n-2} + \frac{6}{11}h_n x'_{n+1}$$

这时,式(5.2.20)的 $p=3$, $a_1=\frac{8}{11}$, $a_2=-\frac{9}{11}$, $a_3=\frac{2}{11}$, $b_0=\frac{6}{11}$,其它系数为零。它需已知 t_n, t_{n-1}和 t_{n-2}时刻的 x 值,才能求出 x_{n+1}。

5.3 储能元件的瞬态离散化模型

在 5.1 节介绍瞬态分析实例时,都是先列出微分方程组,然后利用数值积分公式将微分方程转换为代数方程求解。但在实际编程时,一般采用瞬态离散化模型的方法,即先用数值积分公式推导出储能元件的离散化瞬态模型,然后利用这些模型,在各时刻点上直接建立节点法的代数(差分)方程。这种方法直观、简单,便于计算机处理。下面我们分别介绍电容、电感和互感采用向后欧拉法和梯形法的离散化电路模型。

5.3.1 电容的离散化电路模型

线性电容的支路方程为

$$I = C\frac{dU}{dt}$$

在 t_{n+1} 点,可写为

$$I_{n+1} = CU'_{n+1} \tag{5.3.1}$$

1. 采用向后欧拉法

向后欧拉公式为

$$U_{n+1} = U_n + hU'_{n+1} \tag{5.3.2}$$

将(5.3.2)式代入(5.3.1)式,有

$$I_{n+1} = \frac{C}{h}(U_{n+1} - U_n) \tag{5.3.3}$$

这是电容的向后欧拉公式,由此可画出如图 5.3.1 所示的电容离散化模型。它表明一个线性电容在 t_{n+1} 时刻可以等效为一个值为 $\frac{C}{h}$ 的电导和一个值为 $\frac{C}{h}U_n$ 的电流源并联。

图 5.3.1 电容的向后欧拉法离散化模型

图 5.3.2 电容的梯形法离散化模型

2. 采用梯形法

梯形公式为

$$U_{n+1} = U_n + \frac{h}{2}(U'_{n+1} + U'_n) \tag{5.3.4}$$

由(5.3.1)式,有 $I_{n+1} = CU'_{n+1}$ 和 $I_n = CU'_n$,代入式(5.3.4),得

$$I_{n+1} = \frac{2C}{h}U_{n+1} - \frac{2C}{h}U_n + I_n \tag{5.3.5}$$

这就是电容的梯形法公式。它表明,一个线性电容在 t_{n+1} 时刻可等效为一个值为 $\frac{2C}{h}$ 的电导和一个值为 $\frac{2C}{h}U_n + I_n$ 的电流源并联,如图 5.3.2 所示。

5.3.2 电感的离散化电路模型

线性电感的支路方程是

$$U = L\frac{dI}{dt}$$

在 t_{n+1} 时刻,可写为

$$I'_{n+1} = \frac{U_{n+1}}{L} \tag{5.3.6}$$

1. 采用向后欧拉法

向后欧拉公式为

$$I_{n+1} = I_n + hI'_{n+1} \tag{5.3.7}$$

将(5.3.6)式代入(5.3.7)式,有

$$I_{n+1} = I_n + \frac{h}{L}U_{n+1} \tag{5.3.8}$$

这就是电感的向后欧拉公式，它表示一个线性电感在 t_{n+1} 时刻可以等效为一个值为 $\frac{h}{L}$ 的电导和一个值为 I_n 的电流源并联，如图 5.3.3 所示。

图 5.3.3　电感的向后欧拉法离散化模型

图 5.3.4　电感的梯形法离散化模型

2. 采用梯形法

梯形公式为

$$I_{n+1} = I_n + \frac{h}{2}(I'_{n+1} + I'_n) \tag{5.3.9}$$

由(5.3.6)式可得到 $I'_{n+1} = \frac{U_{n+1}}{L}$ 和 $I'_n = \frac{U_n}{L}$，代入(5.3.9)式，有

$$I_{n+1} = I_n + \frac{h}{2L}U_n + \frac{h}{2L}U_{n+1} \tag{5.3.10}$$

这就是电感的梯形公式，其模型形式如图 5.3.4 所示。它表明，一个线性电感在 t_{n+1} 点可等效为一个值为 $\frac{h}{2L}$ 的电导和一个值为 $\frac{h}{2L}U_n + I_n$ 的电流源并联。

5.3.3　互感的离散化电路模型

对于有 m 个互相通过互感耦合的线圈的特性方程为

$$\begin{aligned}
U_1 &= L_1\frac{dI_1}{dt} + M_{12}\frac{dI_2}{dt} + M_{13}\frac{dI_3}{dt} + \cdots + M_{1m}\frac{dI_m}{dt}\\
U_2 &= M_{21}\frac{dI_1}{dt} + L_2\frac{dI_2}{dt} + M_{23}\frac{dI_3}{dt} + \cdots + M_{2m}\frac{dI_m}{dt}\\
&\cdots\cdots\\
U_m &= M_{m1}\frac{dI_1}{dt} + M_{m2}\frac{dI_2}{dt} + M_{m3}\frac{dI_3}{dt} + \cdots + L_m\frac{dI_m}{dt}
\end{aligned}$$

或写成矩阵形式：

$$U = LI' \tag{5.3.11}$$

其中互感矩阵 L 为

$$L = \begin{bmatrix} L_1 & M_{12} & \cdots & M_{1m} \\ M_{21} & L_2 & \cdots & M_{2m} \\ \vdots & \vdots & \vdots & \vdots \\ M_{m1} & M_{m2} & \cdots & L_m \end{bmatrix}$$

在 t_{n+1} 点我们习惯将(5.3.11)式写成如下矩阵形式:

$$I'_{n+1} = L^{-1}U_{n+1} \tag{5.3.12}$$

由此式可推导出互感的瞬态离散化模型。

1. 采用向后欧拉法

向后欧拉法公式为

$$I_{n+1} = I_n + hI'_{n+1} \tag{5.3.13}$$

将(5.3.12)式代入上式,得

$$I_{n+1} = I_n + hL^{-1}U_{n+1}$$

令矩阵 $G = hL^{-1}$,则

$$I_{n+1} = I_n + GU_{n+1} \tag{5.3.14}$$

即为 m 个相互有互感的线圈的向后欧拉公式。其中第 j 个线圈的公式为

$$I_{j,n+1} = I_{j,n} + g_{j1}U_{1,n+1} + g_{j2}U_{2,n+1} + \cdots g_{jj}U_{j,n+1} + \cdots + g_{jm}U_{m,n+1} \tag{5.3.15}$$

该公式所对应的第 j 个线圈的离散化模型如图 5.3.5 所示。其中 g_{ji} 是 G 矩阵中第 j 行第 i 列的元素,$i = 1, 2, \cdots, m$, $i \neq j$。它表示在 t_{n+1} 时刻互感线圈组中第 j 个线圈可以等效为一个值为 $I_{j,n}$ 的电流源和一个值为 g_{jj} 的电导,以及 m-1 个值为 g_{ji}($i \neq j$)的压控电流源并联。

图 5.3.5 第 j 个线圈的向后欧拉法离散化模型

2. 采用梯形法

梯形法公式为

$$I_{n+1} = I_n + \frac{h}{2}(I'_{n+1} + I'_n) \tag{5.3.16}$$

由(5.3.12)式有 $I'_{n+1} = L^{-1}U_{n+1}$, $I'_n = L^{-1}U_n$,代入(5.3.16)式,得

$$I_{n+1} = I_n + \frac{h}{2}L^{-1}U_n + \frac{h}{2}L^{-1}U_{n+1}$$

令矩阵 $G_T = \frac{h}{2}L^{-1}$, $I_{T,n} = I_n + G_T U_n$,则有

$$I_{n+1} = I_{T,n} + G_T U_{n+1} \tag{5.3.17}$$

这就是 m 个线圈之间有互感耦合的梯形公式。其中第 j 个线圈的公式为

$$I_{j,n+1} = I_{Tj,n} + g_{Tj1}U_{1,n+1} + g_{Tj2}U_{2,n+1} + \cdots + g_{Tjj}U_{j,n+1} + \cdots + g_{Tjm}U_{m,n+1} \tag{5.3.18}$$

它表示第 j 个线圈可等效为一个值为 $I_{Tj,n}$ 的电流源和一个值为 g_{Tjj} 的电导,以及 m-1 个

压控电流源并联组成。其中电流源

$$I_{Tj,n} = I_{j,n} + \sum_{k=0}^{m} g_{Tjk} U_{k,n} \tag{5.3.19}$$

这一节中我们介绍了三种储能元件: 电容、电感和互感, 采用向后欧拉法和梯形法 3 建立了它们的瞬态离散化模型。当应用其它数值积分方法(包括高阶积分法)时, 读者可按照上述方式推导出储能元件相应的离散化模型。

5.4 局部截断误差与稳定性

5.4.1 局部截断误差的计算

在瞬态分析过程中, 每一步都应计算局部截断误差 E_T 是否满足要求, 以便控制步长的选取。计算局部截断误差一般有下面两种方法。

1. 均差法

对 k 阶数值积分公式, 计算其局部截断误差时要估算解向量 x 的 k+ 1 阶导数值。如向后欧拉法的局部截断误差 $E_T = -\frac{h^2}{2}x''(\xi)$, 梯形法的 $E_T = -\frac{h^3}{12}x'''(\xi)$, 分别要计算出 x″ 和 x‴。我们可以用解 $x_{n+1}, x_n, \cdots, x_{n-k}$ 去拟合一个 k+ 1 阶多项式来估算导数值。例如, 对向后欧拉法可用下面的二次多项式:

$$y(t) = a_0 + a_1(t_{n+1} - t) + a_2(t_{n+1} - t)^2 \tag{5.4.1}$$

(5.4.1)式应满足 $x_{n+1} = y(t_{n+1}), x_n = y(t_n), x_{n-1} = y(t_{n-1})$, 由此构成下面的方程组:

$$\begin{bmatrix} x_{n+1} \\ x_n \\ x_{n-1} \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 \\ 1 & h_n & h_n^2 \\ 1 & h_n + h_{n-1} & (h_n + h_{n-1})^2 \end{bmatrix} \begin{bmatrix} a_0 \\ a_1 \\ a_2 \end{bmatrix} \tag{5.4.2}$$

式(5.4.1)的二阶导数值为 $2a_2$。由式(5.4.2)可以求得 a_2, 则二阶导数 x″ 为

$$x''(\xi) \approx 2a_2 = 2\,\frac{\dfrac{x_{n+1} - x_n}{h_n} - \dfrac{x_n - x_{n-1}}{h_{n-1}}}{h_n + h_{n-1}} \tag{5.4.3}$$

同理, 梯形法可以用 $x_{n+1}, x_n, x_{n-1}, x_{n-2}$ 拟合一个三次多项式, 然后求得 x‴(ξ) 的近似值, 即

$$x'''(\xi) = 6\,\frac{\dfrac{\dfrac{x_{n+1} - x_n}{h_n} - \dfrac{x_n - x_{n-1}}{h_{n-1}}}{h_n + h_{n-1}} - \dfrac{\dfrac{x_n - x_{n-1}}{h_{n-1}} - \dfrac{x_{n-1} - x_{n-2}}{h_{n-2}}}{h_{n-1} + h_{n-2}}}{h_n + h_{n-1} + h_{n-2}} \tag{5.4.4}$$

写成一般式为

$$\frac{d^k x}{dt^k} = k!\,DD_k \tag{5.4.5}$$

其中 DD_k 为 k 阶均差。k 阶均差可递归写成

$$DD_k = \frac{DD_{k-1}(t_{n+1}) - DD_{k-1}(t_n)}{\sum_{i=1}^{k} h_{n-i+1}} \tag{5.4.6}$$

一阶均差定义为

$$DD_1 = \frac{x_{n+1} - x_n}{h_n} \tag{5.4.7}$$

应用式(5.4.5)和式(5.4.6),可以计算每个时间点上的局部截断误差。

2. 预估校正法

应用隐式积分公式进行瞬态计算时,常常在每一步先用一个显式公式预估,以显式公式的解作为隐式公式迭代的初值,然后再用隐式公式来求解非线性方程将其解作为校正值,我们称之为预估校正法。通过预估值和校正值还可以估算出局部截断误差值。一般预估公式和校正公式应采用同阶公式,才能较方便地估算出局部截断误差。下面以向后欧拉法和梯形法为例,说明预估校正算法的基本原理。

(1) 一阶预估校正

我们选用向前欧拉法作为预估公式,向后欧拉法作为校正公式。

预估公式:
$$x_{n+1}^{p} = x_n + hx_n' \tag{5.4.8}$$

$$E_T^{p} = x(t_{n+1}) - x_{n+1}^{p} = \frac{1}{2}h^2 x''(\xi) \qquad t_n < \xi < t_{n+1} \tag{5.4.9}$$

校正公式:
$$x_{n+1}^{c} = x_n + hx_{n+1}' \tag{5.4.10}$$

$$E_T^{c} = x(t_{n+1}) - x_{n+1}^{c} = -\frac{1}{2}h^2 x''(\xi) \qquad t_n < \xi < t_{n+1} \tag{5.4.11}$$

用(5.4.9)式减去(5.4.11)式,并设 $\xi = \xi = \xi$ 则

$$E_T^{p} - E_T^{c} = x_{n+1}^{c} - x_{n+1}^{p} = h^2 x''(\xi) \tag{5.4.12}$$

t_{n+1} 时刻的局部截断误差应为校正公式(向后欧拉公式)的误差,即

$$E_T = -\frac{1}{2}h^2 x''(\xi)$$

$$= -\frac{1}{2}(x_{n+1}^{c} - x_{n-1}^{p}) \tag{5.4.13}$$

这就表明,只要计算出 x_{n+1}^{p} 和 x_{n+1}^{c},不必求导数,就可以计算出 E_T 值。

(2) 二阶预估校正

选用台劳公式前三项作预估公式,梯形公式作为校正公式。

预估公式:
$$x_{n+1}^{p} = x_n + hx_n' + \frac{h^2}{2}x_n'' \tag{5.4.14}$$

$$E_T^{p} = x(t_{n+1}) - x_{n+1}^{p} = \frac{h^3}{6}x'''(\xi) \tag{5.4.15}$$

校正公式:
$$x_{n+1}^{c} = x_n + \frac{h}{2}(x_{n+1}' + x_n') \tag{5.4.16}$$

$$E_T^{c} = x(t_{n+1}) - x_{n+1}^{c} = -\frac{h^3}{12}x'''(\xi) \tag{5.4.17}$$

式(5.4.15)和式(5.4.17)相减,并设 $\xi = \xi = \xi$ 则有

$$E_T^{p} - E_T^{c} = x_{n+1}^{c} - x_{n+1}^{p} = \frac{1}{4}h^3 x'''(\xi)$$

所以

$$E_T = -\frac{h^3}{12}x\quad(\xi = -\frac{1}{3}(x^c_{n+1} - x^p_{n+1}) \tag{5.4.18}$$

采用预估校正法是用预估和校正公式的解来求局部截断误差,因而避免了求导数的麻烦。除此以外,它还可以减少非线性迭代的次数,改善收敛情况。因为 N-R 迭代的收敛取决于初值与真值解的接近程度。通常是用前一点的解 x_n 值作为 x_{n+1}的初值。在预估校正算法中,用显式公式求出的解 x^p_{n+1}作 x_{n+1}的初值,显然更接近于真实解,有助于迭代收敛。

5.4.2 变步长策略

在瞬态分析中,步长的变化要受以下几个条件的约束:(1)满足所用数值积分方法的稳定性的要求;(2)满足局部截断误差的要求;(3)保证非线性迭代能够收敛。在满足这些条件的基础上,尽量选择较大的步长,以减少总的积分步数。

下面介绍两种常用的控制步长的方法。

1. 迭代次数控制步长

瞬态分析中,每一时间点上都要进行 N-R 迭代,收敛后才能计算下一个时间点。N-R 迭代能否收敛,收敛的快慢,与步长有很大关系。用迭代次数控制步长虽然是一种“事后处理”的被动方法,但它是一种很实际的、实用程序中常用的一种控制步长方法。

迭代次数控制步长方法的框图如图 5.4.1 所示。给定两个迭代次数的上下界 k_{max}和 k_{min}。若当前 N-R 迭代次数超过 k_{max}次,则认为所用积分步长 h 太大,缩小 8 倍重新进行迭代。若本次步长的 N-R 迭代收敛,且迭代次数小于 k_{min},则认为下一积分步长可以扩大,例如放大 2 倍。这里的 k_{max}, k_{min}和“8 倍”、“2 倍”都是经验常数,一般可取 k_{max} = 10, k_{min} = 5。

图 5.4.1 迭代次数控制步长算法框图

这种变步长的控制方法可以在解变化剧烈的时候,通过缩小步长保证收敛;而当解变

化不大时,又能放大步长以缩短计算时间。

2. 利用截断误差估计步长

在瞬态分析时,对步长的选择,一般是先给定一个初始步长,然后根据对截断误差的估算来判断步长的选择是否合适。若截断误差大于允许值,则应减少步长重新计算;否则,保持原步长进行下一时刻点的计算。但是,这种做法属于事后校正,有时需要多次减少步长才能达到要求。这里介绍一种预估步长的方法,即利用局部截断误差估计下一次步长的方法。

以向后欧拉法为例,其局部截断误差 $E_T = -\frac{h^2}{2}x''(\xi)$。假定整个分析的时间域 T 内的允许误差是 E_{max},则每一步允许的最大误差为

$$e_{max} = \frac{E_{max}}{T}h \tag{5.4.19}$$

设本次步长为 h_n,下一步长为 h_{n+1},令

$$\frac{h_{n+1}}{h_n} = \alpha$$

则本次步长内的局部截断误差为

$$E_{T,n} = -\frac{h_n^2}{2}x''(\xi)$$

下一次步长截断误差为

$$E_{T,n+1} = -\frac{h_{n+1}^2}{2}x''(\xi_{+1}) = -\frac{\alpha^2 h_n^2}{2}x''(\xi_{+1})$$

要求 $|E_{T,n+1}| \le e_{max}$,则

$$\left|\frac{\alpha^2 h_n^2}{2}x''(\xi_{+1})\right| \le e_{max}$$

即

$$\alpha \le \sqrt{\frac{e_{max}}{\left|\frac{h_n^2 x''(\xi_{+1})}{2}\right|}}$$

若近似认为 $\xi_{+1} \approx \xi$,则

$$\alpha \le \sqrt{\frac{e_{max}}{|E_{T,n}|}}$$

或写为

$$h_{n+1} \le \sqrt{\frac{e_{max}}{|E_{T,n}|}}h_n \tag{5.4.20}$$

当本步积分完成后,计算出 $E_{T,n}$,然后利用(5.4.20)式估算出下一步长 h_{n+1}。考虑到 $E_{T,n}$ 是近似值,为保守一些,取

$$h_{n+1} = C\sqrt{\alpha}h_n$$

一般取 C= 0.8～0.9。

再以梯形法为例,其局部截断误差为

$$E_{T,n} = -\frac{h_n^3}{12}x'''(\xi)$$

下一次步长时局部截断误差为

$$E_{T,n+1} = -\frac{h_{n+1}^3}{12}x\ (\xi_{n+1}) = -\frac{\alpha h_n^3}{12}x\ (\xi_{n+1})$$

要求 $|E_{T,n+1}| \leq e_{max}$, 则

$$\left| -\frac{\alpha h_n^3}{12}x\ (\xi_{n+1}) \right| \leq e_{max}$$

即

$$\alpha \leq \left(\frac{e_{max}}{E_{T,n}} \right)^{1/3}$$

或写为

$$h_{n+1} \leq C \left(\frac{e_{max}}{E_{T,n}} \right)^{1/3} h_n \qquad (5.4.21)$$

对其它数值积分方法也按此方法推导出类似的步长估计公式。对于 k 阶积分公式, 估计步长的一般公式为

$$h_{n+1} \leq C \left(\frac{e_{max}}{|E_{T,n}|} \right)^{\frac{1}{k+1}} h_n \qquad (5.4.22)$$

5.4.3 起步与导数不连续点的处理

1. 瞬态分析的起步

设微分方程为

$$x' = f(x, t)$$

其初值条件为

$$x(0) = x_0$$

在瞬态分析时。t= 0 时的初始值 x_0 一般由直流分析程序求得, 然后要用数值积分方法由 x_0 求下一时刻的 $x_1, x_2, \cdots$, 开始的几步称为起步。如何起步, 与所采用的数值积分公式有关。

若采用单步积分公式, 例如向前欧拉法、向后欧拉法和梯形法, 只须知道前一步 x_n, x_n'等信息, 就可计算出 x_{n+1}, 即只需利用本身的数值积分公式就可以了, 故单步法可以自行起步。以下进行分析。

(1) 向前欧拉法。积分公式是显式公式: $x_1 = x_0 + hx_0'$, x_0 和 x_0'都为已知, 代入即可求出 x_1, 故向前欧拉法能够自行起步。

(2) 向后欧拉法。积分公式为 $x_1 = x_0 + hx_1'$, 因为是隐式公式, 已知 x_0 以后, 要经过迭代才能求解出 x_1。因此我们说向后欧拉法也是能自行起步的。

(3) 梯形法。积分公式为 $x_1 = x_0 + \frac{h}{2}(x_0' + x_1')$, 与向后欧拉法相同, 若 x_0, x_0'已知, 经过迭代也可自行起步求出 x_1。

如果积分公式是多步法, 例如 Gear 法的三阶以上公式, 必须知道前几个时刻点的 $x_n, x_{n-1}, x_{n-2}, \cdots$的值, 才能求出 x_{n+1}。因此 $x_1, x_2, \cdots$等前几个时刻点的值必须借助于单步法公式求得。也就是说多步法不能自行起步。

三阶 Gera 公式:

$$x_{n+1} = \frac{18}{11}x_n - \frac{9}{11}x_{n-1} + \frac{2}{11}x_{n-2} + \frac{6}{11}hx_{n+1}'$$

只有当已知了 x_2, x_1, x_0 以后，才能用三阶 Gear 公式求解出 x_3 的值。因此 x_1, x_2 必须先由单步法公式求出。也就是说，一个 k 阶方法必须在前面已求出 k 个解后才能使用。

2. 导数不连续的处理

在瞬态分析过程中，若在某时刻点 t_n，函数的导数 x_n' 不连续，则需选用适当的数值积分方法求解 x_{n+1}，否则会造成解的误差加大。

以梯形法求电容上的电压为例，若电容 C 上的电压 U_C 波形如图 5.4.2 所示，在 t_n 时刻 U_C 波形有间断点，导数 U_C' 不连续。梯形法公式为

$$U_{n+1} = U_n + \frac{h}{2}(U'_{n+1} + U'_n)$$

在求出 U_n 及 U_n' 两个量之后才能通过迭代求出 U_{n+1}。由于 $U_C(t)$ 波形在 t_n 时刻有间断点，U_n' 不连续，即 $U'(t_n^-) \neq U'(t_n^+)$，而在分析过程中求得的是 $U_n' \approx U'(t_n^-)$，所以由此求出的 U_{n+1} 必然不精确。若误差过大，甚至会造成整个瞬态分析过程的失败。

若采用向后欧拉法求解，则不会产生这类问题。因向后欧拉公式 $U_{n+1} = U_n + hU'_{n+1}$，在求 U_{n+1} 时，不需要 U_n' 的值，因此在 t_n 点 U_n' 不连续不会影响 U_{n+1} 的计算精度。

图 5.4.2　函数的导数不连续点示意图

根据以上分析可知，在所有数值积分公式中，凡是要求出 x_n' 才能求出 x_{n+1} 的公式，在由导数不连续点求下一点的 x 值时，都会发生计算值不正确的现象。对这些点必须做特殊处理。一般办法是：

(1) 若已知输入函数中有导数不连续点，则在程序中应预先指定在这些点改用向后欧拉法等适当的积分公式。因为往往输入波形导数不连续会形成输出响应波形导数也不连续的现象。

(2) 若无法预知导数不连续点，例如某些方波振荡电路等，可以采取在难于收敛的点上自动改变积分公式的方法。有时也可用大幅度缩小步长的方法来处理，因实际电路中有时并没有导数不连续点，而是有一些导数值变化剧烈的点。

5.4.4　绝对稳定和 stiff 稳定

通常在选择数值积分方法时，要同时考虑局部截断误差和稳定性两个方面。为了直观地描述积分方法的稳定性，常用在 $h\lambda$ 复平面上的稳定区域来表示，不同的数值积分方法，其稳定域的大小有很大差别。为了判断稳定性的优劣。Dahliquist 提出了一个绝对稳定的概念。如果一个数值积分方法的稳定域包含了整个 $h\lambda$ 平面的左半平面，则称这种积分方法是绝对稳定的。对于绝对稳定的方法，不管取多大的积分步长，当方程本身是稳定时，此方法也是稳定的。按照此标准，前面介绍的向后欧拉法和梯形法都是绝对稳定的，而向前欧拉法则不是绝对稳定的。

绝对稳定的积分方法适合于求解电路方程。因为电路方程中寄生元件与耦合、旁路元

件的时间常数 τ_{min} 与 τ_{max} 之间相差往往很悬殊(等同于方程的两个特征值的负实部 λ_{max} 与 λ_{min} 相差悬殊),可能相差 $10^2 \sim 10^8$ 之多。当常微分方法中最大时间常数与最小时间常数相差达若干量级时,我们称之为 stiff 方程,通常译为刚性、坏条件或病态方程。电路方程多属于刚性方程。如果在解刚性方程时采用稳定性很差的向前欧拉法,其最大允许步长 h_{max} 要受 τ_{min} 的限制($h < 2\tau_{min}$),既使瞬态过程处于缓慢变化阶段,h_{max} 也不能加大。但是,瞬态分析的整个区间 T 一般由 τ_{max} 决定,T 与 τ_{max} 属于同一量级。τ_{min} 与 τ_{max} 相差悬殊,则在整个积分区间内计算的步数就很多,既耗费了运算时间又加大了累积误差。例如,某电路中 $|\lambda_{max}| = 1000$, $|\lambda_{min}| = 1$,即 $\tau_{min} = 0.001$, $\tau_{max} = 1$,采用向前欧拉法的稳定性要求是 $h_{max} < 2\tau_{min}$,即 h_{max} 必须小于 0.002s,由 $\tau_{max} = 1s$ 决定取总的积分时间 $T = 5s$,则总的积分步数至少为 5/0.002(等于 2500 步)。如果采用绝对稳定的向后欧拉法或梯形法,h_{max} 就不受 τ_{min} 的限制,选择步长时只要考虑满足局部截断误差要求即可,这样就大大减少了积分步数和累积误差。

Dahliquist 对数值积分方法的稳定性进行了归纳,提出:(1)绝对稳定的数值积分方法没有超过二阶的。(2)在所有二阶方法中,梯形法的局部截断误差最小。(3)任何显式公式都不是绝对稳定的。由此看来,对于电路问题,若要求算法是绝对稳定的,那么可用的数值积分方法寥寥无几。

由于绝对稳定条件排除了许多可用的积分方法。1968 年 Gear 提出了一种 stiff 稳定条件,Gear 认为只要积分方法满足 stiff 条件,既使不是绝对稳定的,也可用来求解刚性方程。stiff 稳定条件的定义是:若存在正常数 D, d, β,使积分公式在 R_1 区$\{Re(h\lambda) \le -D\}$中绝对稳定,而在 R_2 区$\{-D < Re(h\lambda) < d, |Im(h\lambda)| < \beta\}$中对试验方程 $x' = \lambda x$ 是精确的,则称此积分公式是 stiff 稳定的。稳定区域如图 5.4.3 中斜线区域 R_1, R_2 所示。stiff 稳定不属于绝对稳定,但任何绝对稳定的算法显然一定是 stiff 稳定的。

图 5.4.3 stiff 稳定区域

5.5 基尔算法

基尔算法,即 Gear 法是以 stiff 稳定条件为基础的一种精度较高的多步积分方法。按照 5.2 节中介绍的多步法的一般式(5.2.20),即有

$$x_{n+1} = \sum_{i=1}^{p} a_i x_{n-i+1} + h_n \sum_{i=0}^{p} b_i x'_{n-i+1}$$

Gear 法规定该式中除了 $b_0, a_1, a_2, \cdots, a_k$ 取值之外,其它系数均为零,即用过去 k 个点的 $x_n, x_{n-1}, \cdots x_{n-k+1}$ 的值及 x'_{n+1} 来逼近 x_{n+1}。其公式如下:

$$x_{n+1} = a_1 x_n + a_2 x_{n-1} + \cdots + a_k x_{n-k+1} + b_0 h x'_{n+1} \qquad (5.5.1)$$

式中的 k+1 个待定系数可用待定系数法,由下面的 k 阶多项式求出:

$$y(t) = C_0 + C_1(t - t_n) + C_2(t - t_n)^2 + \cdots + C_k(t - t_n)^k \qquad (5.5.2)$$

令 $y(t)$ 通过 $x_n, x_{n-1}, \cdots, x_{n-k+1}$ 点，且在 t_{n+1} 点上的斜率等于 x'_{n+1}，即 $y(t_n)=x_n, y(t_{n-1})=x_{n-1}, \cdots, y(t_{n-k+1})=x_{n-k+1}$，以及 $y'(t_{n+1})=x'_{n+1}$，则可构造出如下矩阵方程组：

$$\begin{bmatrix} 1 & 0 & 0 & \cdots & 0 & C_0 \\ 1 & -1 & (-1)^2 & & (-1)^k & C_1h \\ 1 & -2 & (-2)^2 & \cdots & (-2)^k & C_2h^2 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ 1 & (1-k) & (1-k)^2 & \cdots & (1-k)^k & C_{k-1}h^{k-1} \\ 0 & 1 & 2 & \cdots & k & C_kh^k \end{bmatrix} = \begin{matrix} x_n \\ x_{n-1} \\ x_{n-2} \\ \vdots \\ x_{n-k+1} \\ hx'_{n+1} \end{matrix} \tag{5.5.3}$$

式(5.5.1)写成矩阵形式为

$$x_{n+1} = [a_1, a_2, \cdots, a_k, b_0] \begin{matrix} x_n \\ x_{n-1} \\ \vdots \\ x_{n-k+1} \\ hx'_{n+1} \end{matrix} \tag{5.5.4}$$

将(5.5.3)式代入(5.5.4)式，得

$$x_{n+1} = [a_1, a_2, \cdots, a_k, b_0] \begin{bmatrix} 1 & 0 & 0 & \cdots & 0 & C_0 \\ 1 & -1 & (-1)^2 & \cdots & (-1)^k & C_1h \\ 1 & -2 & (-2)^2 & \cdots & (-2)^k & C_2h^2 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ 1 & (1-k) & (1-k)^2 & \cdots & (1-k)^k & C_{k-1}h^{k-1} \\ 0 & 1 & 2 & \cdots & k & C_kh^k \end{bmatrix} \tag{5.5.5}$$

而由 $x_{n+1}=y(t_{n+1})$，有

$$x_{n+1} = C_0 + C_1h + C_2h^2 + \cdots + C_kh^k$$

或写为

$$x_{n+1} = [1, 1, \cdots, 1] \begin{matrix} C_0 \\ C_1h \\ \vdots \\ C_{k-1}h^{k-1} \\ C_kh^k \end{matrix} \tag{5.5.6}$$

比较(5.5.5)和(5.5.6)，可得

$$[a_1, a_2, \cdots, a_k, b_0] \begin{bmatrix} 1 & 0 & 0 & \cdots & 0 \\ 1 & -1 & (-1)^2 & \cdots & (-1)^k \\ 1 & -2 & (-2)^2 & \cdots & (-2)^k \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ 1 & (1-k) & (1-k)^2 & \cdots & (1-k)^k \\ 0 & 1 & 2 & \cdots & k \end{bmatrix} = \begin{matrix} 1 \\ 1 \\ \vdots \\ 1 \\ 1 \end{matrix}$$

将上式转置,得

$$\begin{matrix} 1 & 1 & 1 & \cdots & 1 & 0 & a_1 & & 1 \\ 0 & -1 & -2 & \cdots & 1-k & 1 & a_2 & & 1 \\ 0 & (-1)^2 & (-2)^2 & \cdots & (1-k)^2 & 2 & \vdots & = & \vdots \\ \vdots & \vdots & \vdots & & \vdots & \vdots & a_k & & 1 \\ 0 & (-1)^k & (-2)^k & \cdots & (1-k)^k & k & b_0 & & 1 \end{matrix} \qquad (5.5.7)$$

由(5.5.7)式取不同的 k 值,便可得到各阶 Gear 公式。

例如取 k= 1,由

$$\begin{matrix} 1 & 0 & a_1 \\ 0 & 1 & b_0 \end{matrix} = \begin{matrix} 1 \\ 1 \end{matrix}$$

解出 $a_1=1$, $b_0=1$,即

$$x_{n+1} = x_n + hx'_{n+1}$$

这就是一阶基尔公式,即向后欧拉公式。

若取 k= 2,由

$$\begin{matrix} 1 & 1 & 0 & a_1 \\ 0 & 1 & -1 & a_2 \\ 0 & 1 & 2 & b_0 \end{matrix} = \begin{matrix} 1 \\ 1 \\ 1 \end{matrix}$$

解出 $a_1=\frac{4}{3}$, $a_2=-\frac{1}{3}$, $b_0=\frac{2}{3}$,即

$$x_{n+1} = \frac{4}{3}x_n - \frac{1}{3}x_{n-1} + \frac{2}{3}hx'_{n+1}$$

这就是二阶基尔公式。

下面列出,取 k= 1～6 时由(5.5.7)式求解出的 6 个 Gear 公式:

一阶公式:

$$x_{n+1} = x_n + hx'_{n+1} - \frac{1}{2}h^2x''(\xi)$$

二阶公式:

$$x_{n+1} = \frac{4}{3}x_n - \frac{1}{3}x_{n-1} + \frac{2}{3}hx'_{n+1} - \frac{2}{9}h^3x'''(\xi)$$

三阶公式:

$$x_{n+1} = \frac{18}{11}x_n - \frac{9}{11}x_{n-1} + \frac{2}{11}x_{n-2} + \frac{6}{11}hx'_{n+1} - \frac{3}{22}h^4x^{(4)}(\xi)$$

四阶公式:

$$x_{n+1} = \frac{48}{25}x_n - \frac{36}{25}x_{n-1} + \frac{16}{25}x_{n-2} - \frac{3}{25}x_{n-3} + \frac{12}{25}hx'_{n+1} - \frac{12}{125}h^5x^{(5)}(\xi)$$

五阶公式:

$$x_{n+1} = \frac{300}{137}x_n - \frac{300}{137}x_{n-1} + \frac{200}{137}x_{n-2} - \frac{25}{137}x_{n-3} + \frac{12}{137}x_{n-4} + \frac{60}{137}hx'_{n+1} - \frac{10}{137}h^6x^{(6)}(\xi)$$

六阶公式:

$$x_{n+1} = \frac{360}{147}x_n - \frac{450}{147}x_{n-1} + \frac{400}{147}x_{n-2} - \frac{225}{147}x_{n-3} + \frac{72}{147}x_{n-4} - \frac{10}{147}x_{n-5}$$

$$+\frac{60}{147}hx'_{n+1}-\frac{60}{1029}h^7x^{(7)}(\xi) \tag{5.5.8}$$

6 个 Gear 公式的稳定区域如图 5.5.1 中阴影区域所示。只有一阶和二阶 Gear 公式是绝对稳定的,其它所有 Gear 公式都只是 stiff 稳定的。

图 5.5.1　一至六阶 Gear 公式的稳定域

大于一阶的 Gear 公式都属于多步法,在采用 k 阶 Gear 公式计算 x_{n+1} 时,需已知 x_n 及 k- 1 个过去时刻的解(x_{n-1}, …, x_{n-k+1})。因此,如同任何一个多步法一样,需对这些时刻点上的解进行存储和处理工作。另外,二阶以上的 Gear 公式不能自行起步,即一个 k 阶方法必须在前面已求得 k 个解后才能使用。对 Gear 法,必须先用一阶,然后才能使用二阶……在求得 6 个时间点上的解以后,一至六阶的 Gear 公式就都可以使用了。

一个通用的 Gear 积分方法,还应包括对阶数和步长的自动选择,及对过去时刻点的值的计算方法等方面的问题,读者可参阅有关文献,在此我们就不赘述了。

5.6　瞬态分析程序及分析实例

5.6.1　瞬态分析程序介绍

一个基本的瞬态分析程序流程图如图 5.6.1 所示。其特点是:

(1) 可以用直流非线性分析结果作为初始状态,也可以由用户输入初始状态直接进

图 5.6.1　瞬态分析程序流程图

行瞬态分析。

(2) 第一、二时刻点或波形间断点采用向后欧拉法计算,其它各时刻均采用梯形法计算。

(3) 当非线性 N-R 迭代次数达到最大允许次数 k_{max} 仍不收敛,或局部截断误差大于允许值,均采取缩小步长的措施。

(4) 利用当前步的截断误差预估下一步长。

(5) 采用数值积分方法(向后欧拉法或梯形法)处理电容、电感或互感模型,采用 N-R 方法处理非线性器件模型,都在“建立电路方程组”的程序模块中实现。

瞬态分析是求解电路的时域响应,它是电路 CAD 软件中非常重要而且非常有用的一种分析。由于瞬态分析是一种最细致的波形分析方法,为了保证计算的精度、速度和收敛性,实际程序的实现和应用都需要很多的技术和技巧,比图 5.6.1 的内容要复杂得多。

在 SPICE 程序中瞬态分析语句是:

.TRAN TSTEP TSTOP < TSTART < TMAX> > < UIC>

其中 TSTEP 是打印或绘图输出的时间步长, TSTOP 是分析终止时间, TSTART 是输出的起始时间。实际上瞬态分析总是从零时刻开始, 只是在零到 TSTART 时间区间内没有输出, 而且分析结果也没有被存储。TMAX 是允许的最大步长, 它的缺省值是 min{TSTEP, (TSTOP- TSTART)/50}。为了保证计算精度, 正确的设置 TMAX 值是十分重要的。UIC (Use Initial Conditions)是一个任选的关键字, 它表示采用用户所指定的初始条件进行瞬态分析, 而不首先调用直流分析来确定瞬态的初始条件。用户指定的初始条件是在晶体管的端口或电容、电感两端用 IC=×× 设置初始电压或电流来确定, 也可以用.IC 语句设置任何带节点的电位值来确定。

傅立叶分析是以瞬态分析为基础的一种谐波分析方法。它对瞬态分析结果的最后一个周期波形数据进行抽样, 得到直流、基波分量和第 2 到第 9 次谐波的分量, 并计算出失真度值。傅立叶分析的典型语句是:

```
.FOUR 100K V(2)
```

其中 100K 是基波频率, V(2)是输出变量, 即对 V(2)的输出波形进行傅立叶分析, 可计算出各次谐波分量的大小。

5.6.2 分析实例

1. 考比兹振荡器

图 5.6.2 所示是一个考比兹(Colpitts)振荡电路, 振荡周期为 6.5MHz。下面我们对该电路进行瞬态分析, 观察振荡器的起振过程, 并测试它的振荡频率。

图 5.6.2 考比兹振荡电路

电路输入网单文件如下:

```
Colpitts  Oscillator
R1  1  5  12K
R2  1  0  4.7K
R3  3  0  1K
R4  2  4  10K
```

```
* R4  2  4  RM  1K
R5  4  5  100
C1  2  3  100P
C2  3  4  1000P
C3  1  0  0.1U
C4  4  0  0.1U
L1  2  6  6.595U
R6  4  6  0.1
Q1  2  1  3  MQ      ;图 5.6.2 中为 T1
VCC  5  0  12
.MODEL  MQ  NPN  IS= 1E-14  RB= 10  BF= 100
+  CJE= 1P  CJC= 1P  TF= 1N
*.MODEL  RM  RES(R= 1)
.OP
.TRAN  1E-8  10E-6  0  5E-8
*.STEP  RES  RM(R)  LIST  100  1K  10K
*.STEP  DEC  NPN  MQ(TF)  0.1N  10N  1
*.STEP  NPN  MQ(CJC)  LIST  1P  20P
*.FOUR  6.5MEG  V(2)
.OPTIONS  ITL5= 0
.PROBE
.END
```

输出振荡波形如图 5.6.3 所示。

图 5.6.3　振荡器振荡波形 V(2)

用 PSpice 还可以对该电路进行下面几种分析:

(1) 通过改变负载电阻的 R_4 的值: 100Ω, 1k, 10k, 观察输出振荡波形幅度的变化。在 PSpice 输入文件中需设置电阻模型和扫描语句, 即

```
R4  2  4  RM  1K
.MODEL  RM  RES(R= 1)
```

.STEP RES RM(R) LIST 100 1K 10K

(2) 通过改变晶体管参数;如 CJC= 1～20PF, TF= 0.1～10NS, 观察对振荡电路起振点的影响。在 PSpice 输入文件中,增加的扫描语句为:

.STEP NPN MQ(CJC) LIST 1P 20P

.STEP DEC NPN MQ(TF) 0.1N 10N 1

(3) 对输出振荡波形做傅立叶分析。

另外,振荡电路的瞬态分析还有一些小技巧:

(1) 在晶体管的基极或电源 V_{CC} 上加一激励信号,则电路能够很快起振,减少起振过程。例如:

VCC 6 0 PULSE (13 12 2NS)

(2) 在瞬态分析中应适当地设置分析语句中的最大步长 TMAX 值,例如:

.TRAN 1E-8 1E-5 0 5E-8

否则,计算精度会受影响,电路可能停振或难于起振。

2. 多谐振荡器

图 5.6.4 是一个简单的多谐振荡器,振荡周期约为 90kHz。对该电路进行瞬态分析时,需要用.IC 语句设置 T_1 和 T_2 的初始状态:如 T_1 导通, T_2 截止,这样电路才能正常翻转。

图 5.6.4 多谐振荡器

图 5.6.5 V(4)输出波形

输入数据文件:

```
BJT abtable multivibrater
Q1  1  2  0  MD1    ;图 5.6.4 中为 T1
Q2  4  5  0  MD1    ;图 5.6.4 中为 T2
RC1  1  7  1K
RC2  4  7  1K
RB1  2  7  12K
RB2  5  7  12K
C1  1  5  640P
C2  2  4  640P
```

```
VCC  7  0  12
.MODEL  MD1  NPN  IS= 1E- 16  BF= 100  TF= 10N
+ CJC= 10P  CJE= 10P
.TRAN  1U  100U  0  1U  UIC
.IC  V(1)= 0.0  V(4)= 12
.OPTIONS  ITL5= 0
.PROBE
.END
```

输出振荡波形 V(4) 如图 5.6.5 所示。

习　　题

5.1　采用向后欧拉法建立题图 5.1 所示电路在 t_n 时刻的电路方程。

题图　5.1

5.2　假定在 t_n 时刻，题图 5.2 所示电路中的 V_{1n}, V_{2n}, V_{3n}, I_{En} 及电感中的电流 I_{1n}, I_{2n}, I_{3n} 都已知，请列出此电路在 t_{n+1} 时刻的电路方程。其中电感、电容、互感采用向后欧拉法的离散化模型。

题图　5.2

题图　5.3

5.3　采用梯形法，写出在 t_{n+1} 时刻题图 5.3 所示电路的瞬态方程。

5.4　在题图 5.4 所示电路中，电容两端的初始电压 $U_C(0)=0$，试用下列 3 种方法求

在 [0, 3] s 时间区间内, U_C 的瞬态响应。设步长 $h = 0.15s$。

(1) 采用向后欧拉法;

(2) 采用梯形法;

(3) 采用解析法。

然后以解析法的解为精确解, 比较向后欧拉法与梯形法的精度。

题图 5.4

5.5 用 PSpice 程序计算习题 4.7 所示电路的瞬态响应。

(1) 在 f = 2kHz 时, 计算电路的最大不失真功率(失真度小于 5%)。

(2) 分析在什么情况下可能出现交越失真现象, 并绘出失真波形。

5.6 用 PSpice 程序计算习题 4.8 中电路的瞬态响应, 其 f = 5kHz。

(1) 调整负载电阻之值, 观察输出对管 T_3, T_4 集电极电流波形的变化。

(2) 改变输出对管 T_3, T_4 的模型参数: IS, RB, RC, RE, VAF 等等, 观察两管集电极电流的变化, 并说明原因。

(3) 计算最大不失真功率(失真度小于 5%)。

5.7 用 PSpice 程序计算题图 5.7 所示的 LC 振荡器的瞬态响应。

题图 5.7

(1) 计算静态工作点。晶体管参数有: I_S, β_F, τ, C_{JEo}, C_{JCo}, U_A 等等。

(2) 通过加激励信号, 加速电路起振过程, 并测量振荡频率。

(3) 改变负载电阻和反馈系数, 观察其对振荡幅度的影响。

5.8 以 PSpice 程序为设计工具, 设计一个“西勒”电容反馈三点式 LC 振荡器, 要求:

(1) $f_0 = 1MHz$;

(2) 电源功耗小于 35mW;

(3) 反馈系数 $F \approx 0.15$。

第 6 章　频 域 分 析

电路的频域分析在电路理论和电路 CAD 领域中都是一种十分重要的分析方法。这种方法的实质是研究在不同频率的正弦激励作用下电路的稳态响应,从而获得电路的频率特性。在这一章中,我们主要介绍频域分析中的交流小信号分析和零极点分析方法。

6.1　交流小信号分析

交流小信号分析是一种频域分析方法。它是研究在小信号输入情况下,电路的电压增益、频率特性等性能。由于电路工作在小信号情况下,电路中非线性器件可以采用其工作点附近的线性化模型,整个电路可以按线性电路来处理。交流小信号分析程序与直流线性分析程序基本相同。所不同的是,在交流小信号分析中电路元件的导纳(或阻抗)是频率的函数,是复数量,各种独立源、信号源也是频率的函数,因而形成的电路方程组是一个复数线性代数方程组。可采用第 2 章介绍的求解复数代数方程组的方法来计算。

6.1.1　元器件的交流小信号模型

在交流小信号分析中,电阻(或电导)和各种受控源的模型基本上与直流分析中的相同,一般情况下它们不是频率的函数。下面我们用节点法和改进节点法描述部分电抗元件与非线性半导体器件的交流小信号模型,及其它们对复数电路方程组的贡献。

1. 电容

电容的导纳为 SC,这里 S= jω。其支路如图 6.1.1 所示,对复数电路方程组的贡献列于表 6.1.1 中。

表 6.1.1　电容送值表

行 \ 列	V_k	V_l	RHS
k	SC	- SC	
l	- SC	SC	

图 6.1.1　电容支路

2. 电感

电感的导纳为 1/SL,其中 S= jω。其支路如图 6.1.2 所示,对复数电路方程组的贡献列于表 6.1.2 中。

表 6.1.2　电感送值表(1)

行 \ 列	V_k	V_l	RHS
k	1/SL	- 1/SL	
l	- 1/SL	1/SL	

图 6.1.2　电感支路

电感支路的电流常常是有用的变量,用改进节点法可求出其支路电流。采用改进节点法列方程时,电感的送值表如表 6.1.3 所示。

表 6.1.3　电感送值表(2)

行＼列	V_k	V_l	I_L	RHS
k			1	
l			- 1	
BR	1	- 1	- SL	

3. 互感

在交流小信号分析中,经常遇到含有线圈之间磁通互相耦合的互感支路。设电路中有两个电感元件 L_1 和 L_2,它们之间的互感为 M($M = M_{12} = M_{21}$),电感两端的电压分别为 U_1 和 U_2,支路电流分别为 I_1 和 I_2,如图 6.1.3 所示。

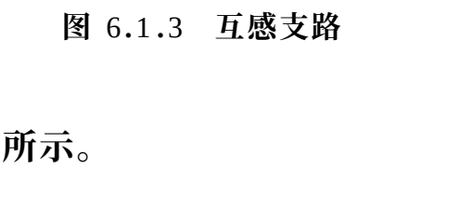


图 6.1.3　互感支路

由 KVL,可以写出

$$\begin{aligned} U_1 &= V_i - V_j = SL_1 I_1 + SMI_2 \\ U_2 &= V_k - V_l = SMI_1 + SL_2 I_2 \end{aligned} \qquad (6.1.1)$$

将互感支路电流 I_1, I_2 作为未知变量,其送值表如表 6.1.4 所示。

表 6.1.4　互感支路送值表

行＼列	V_i	V_j	V_k	V_l	I_1	I_2
i					1	
j					- 1	
k						1
l						- 1
BR1	1	- 1			$- SL_1$	- SM
BR2			1	- 1	- SM	$- SL_2$

如果电路中有多个电感,它们两两之间都存在互感,可按照上面情况类似处理。

4. 理想变压器

理想变压器(见图 6.1.4(a))可以看作是电压控制电压源和电流控制电流源的组合,如图 6.1.4(b)所示。理想变压器支路方程为

$$\begin{aligned} U_2 &= N U_1 \\ I_2 &= -\frac{1}{N} I_1 \end{aligned} \qquad (6.1.2)$$

这里 $N = n_2/n_1$。对于图 6.1.4(b)的变压器模型,式(6.1.2)可写为

$$\begin{aligned} V_k - V_l &= N(V_i - V_j) \\ I_1 &= - N I_2 \end{aligned} \qquad (6.1.3)$$

图 6.1.4 (a) 理想变压器; (b) 理想变压器模型

理想变压器对电路方程的贡献如表 6.1.5 所示。

表 6.1.5 理想变压器送值表

行 \ 列	V_i	V_j	V_k	V_l	I_2
i					- N
j					N
k					1
l					- 1
BR	- N	N	1	- 1	

5. 二极管交流小信号模型

交流小信号分析时, 第 3 章中所介绍的二极管模型(图 3.1.1 亦即图 6.1.5(a))中非线性电流源 I_D 等效为二极管直流工作点附近的跨导。图 6.1.5(b) 中示出了二极管的交流小信号模型, 其中 G_{DM} 和 C_D 值均依赖于直流工作点。

图 6.1.5 (a) 二极管原始模型; (b) 二极管交流小信号模型

二极管的电流与端电压的关系是:

$$I_D = I_S(e^{U_D/V_T} - 1)$$

在直流工作点上, 求等效跨导 G_{DM} 和等效电容 C_D:

$$G_{DM} = \left.\frac{I_D}{U_D}\right|_{工作点} = \frac{I_S}{V_T}e^{U_D/V_T} \tag{6.1.4}$$

$$C_D = \frac{q\tau I_S}{nkT}\exp\frac{qU_D}{nkT} + C_{J0}\left(1 - \frac{U_D}{\phi_B}\right)^{-m} \tag{6.1.5}$$

上式中各参数意义同式(3.1.3)。

二极管交流小信号模型对电路方程组的贡献, 由电阻、电导和电容的贡献组合而成。

读者可自己列出其送值表。

6. 双极型晶体管交流小信号模型

双极型晶体管交流小信号模型是以第 3 章中介绍的各种双极型晶体管基本模型为基础,在直流工作点附近进行线性化而构成的一种线性化模型。例如,双极型晶体管的 EM2 混合 π交流小信号模型如图 6.1.6 所示。图中 $R_{BB'}$, $R_{EE'}$, $R_{CC'}$ 分别为各极体电阻。g_m 是由晶体管直流工作点确定的正向跨导(在正常工作范围内,反向跨导值为零),且

$$g_m = \left.\frac{I_{FC}}{U_{BE'}}\right|_{工作点} \tag{6.1.6}$$

图 6.1.6　晶体管交流小信号模型

其中 I_{FC} 是 EM2 模型中 BE 结二极管电流,有

$$I_{FC} = I_S(e^{U_{BE'}/V_T} - 1)$$

模型中其它元件参数由下面各式表示:

$$R_{BE'} = \frac{\beta}{g_m} \tag{6.1.7}$$

$$C_{BE'} = g_m\tau_f + \frac{C_{JE0}}{1 - \frac{U_{BE'}}{\phi_E}^{m_E}} \tag{6.1.8}$$

$$C_{BC'} \approx \frac{C_{JC0}}{1 - \frac{U_{BC'}}{\phi_C}^{m_C}} \tag{6.1.9}$$

式中各参数意义同第 3 章 EM2 模型。

7. MOS 场效应晶体管交流小信号模型

MOS 场效应晶体管交流小信号模型如图 6.1.7 所示。

模型中电导 G_{BD} 和 G_{BS} 是漏-衬底和源-衬底的 PN 结的等效电导。电导 G_G, g_m, 和 g_{mB} 分别是直流工作点附近的等效跨导:

$$G_G = \left.\frac{I_{DS}}{U_{DS}}\right|_{工作点}$$

$$g_m = \left.\frac{I_{DS}}{U_{GS}}\right|_{工作点}$$

图 6.1.7　MOS 场效应管交流小信号模型

$$g_{mB}=\left.\frac{I_{DS}}{U_{BS}}\right|_{工作点}$$

电容 C_{GD}, C_{GS}, C_{GB}, C_{BD}和 C_{BS}分别为 MOS 直流工作点的函数,其值由第 3 章 MOS 场效应管模型的电容公式(3.3.17)～(3.3.20)决定。

6.1.2　交流小信号分析流程与实例

交流小信号分析属于线性稳态分析,程序首先对电路进行直流分析,计算电路的直流工作点,以确定电路中所有非线性器件的线性化交流小信号模型参数。然后,在用户指定的频率范围内对这个线性化电路进行频域分析。交流小信号分析的结果有:

(1) 电路的幅频和相频特性曲线;

(2) 电路的传输函数:电压增益、传输阻抗等;

(3) 电路的等效输入阻抗和等效输出阻抗;

(4) 等效输入和输出噪声电平。

SPICE 程序可以用交流小信号分析来模拟电路中电阻和半导体器件产生的白噪声。器件的等效噪声源值自动由电路工作点来确定,每个噪声源产生的噪声都在指定的求和点相加。SPICE 程序计算出每个频率点上总的输出噪声电平及其等效输入噪声电平。交流小信号分析程序框图如图 6.1.8 所示。现举一交流分析的实例。

图 6.1.8　交流小信号分析框图

例 6.1.1　一个带通滤波网络如图 6.1.9 所示,用 PSpice 计算该网络的频响特性和输入、输出阻抗。

网络输入网单文件如下:

```
AC Sweep
VI   1   0   AC   1      ;图 6.1.9 中为 EA
R1   1   2   50
C1   2   0   3900P
C2   2   3   3300P
C3   3   0   3900P
L1   2   3   19.1H
R2   3   0   50
.AC   DEC   10   10K   10MEG
.PROBE
.END
```

图 6.1.9　带通滤波网络

计算出的幅频特性如图 6.1.10 所示。计算出的输入阻抗曲线如图 6.1.11 所示。

图 6.1.10　带通滤波网络幅频特性

图 6.1.11　输入阻抗曲线

计算输出阻抗时,应将图 6.1.9 中输入源短路,并在输出端加一个电流源 II 激励。

输入网单文件改为:

```
A FILTER
VI   1   0   AC   0      ;图 6.1.9 中为 EA
II   0   3   AC   1
RI   1   2   50
C1   2   0   3900P
C2   2   3   3300P
C3   3   0   3900P
L1   2   3   19.1H
RL   3   0   50
.AC   DEC   10   10K   10MEG
.PROBE
.END
```

图 6.1.12　输出阻抗曲线

计算出的输出阻抗曲线如图 6.1.12 所示。

例 6.1.2 图 6.1.13 所示的是一个视频放大器电路，我们利用 PSpice 程序中交流小信号分析部分计算这个电路在 1MHz～6MHz 频率区间的幅频响应和相频特性。电路中双极晶体管模型采用图 6.1.6 所示的线性混合 π模型。

图 6.1.13 视频放大器电路

电路输入网单文件是:

```
A   VF   Amplifier
R1   5   2   4.7K
R2   2   0   10K
R3   3   6   4.7K
R4   4   7   51
R5   7   0   1K
R6   7   8   27
R7   11   13   1K
R8   9   0   360
R9   10   12   6.8K
R10   13   0   150K
C1   1   2   33U
C2   7   0   470P
C3   4   8   2200P
C4   8   9   220U
C5   6   11   22U
C6   13   0   10P
L1   3   6   68U
L2   6   10   82U
D1   6   12   DM
Q1   3   2   4   QM        ,图中为 T1
VCC1   5   0   12
VCC2   12   0   100
```

```
VI   1   0   AC   1        ,图中为 EA
.MODEL   DM   D   IS= 1E- 12   RS= 50
.MODEL   QM   NPN   IS= 1E- 13   BF= 80   RB= 10
+    CJC= 2P   CJE= 2P   TF= 2NS
.OP
.AC   LIN   50   1MEG   6MEG
.PROBE
.END
```

计算出的直流工作点如下:

NODE	VOLTAGE	NODE	VOLTAGE	NODE	VOLTAGE	NODE	VOLTAGE
(1)	0.0000	(2)	7.8912	(3)	53.7020	(4)	7.2452
(5)	12.0000	(6)	53.7020	(7)	6.8936	(8)	6.8936
(9)	0.0000	(10)	53.7020	(11)	0.0000	(12)	00.0000
(13)	0.0000						

电压源电流和总的功率损耗如下:

```
VOLTAGE   SOURCE   CURRENTS
NAME              CURRENT

VCC1              - 8.742E- 04
VCC2              - 6.808E- 03

TOTAL   POWER   DISSIPATION   6.91E- 01   WATTS
```

计算出的幅频特性曲线和相频特性曲线如图 6.1.14 所示。

图 6.1.14　视频放大器频响特性曲线

6.2　零极点分析

零极点分析是电路系统中一种十分重要而且有效的分析方法。在复频域(S 域)中,凭借电路的网络函数在复平面零点与极点分布的研究,可以简明、直观地给出电路响应的许

多规律。例如,由电路的网络函数的零极点分布,可以预言给定激励函数的条件下,电路响应的时域特性,便于划分响应的各个分量(自由响应分量与强迫响应分量);而且可以用来描述电路系统的频响特性(幅频特性和相频特性)。零极点分析的另一个重要作用是常常用于判定放大器或反馈系统的稳定性,以及用来说明自激振荡器建立自激振荡的条件。鉴于零极点分析在电路系统中的重要作用,我们在这一节中,从电路 CAD 的角度介绍计算电路网络函数零极点的分析算法。

1. 电路网络函数的零点和极点

设电路方程为

$$TX = B \tag{6.2.1}$$

解向量 X 由节点电位和某些支路电流组成。电路是由电压源或电流源激励的,用 U_I 或 I_I 表示,输出用 U_O 和 I_O 表示。若电路有一输入,我们可以定义以下 6 个网络函数:

$$\begin{aligned}
&\text{输入阻抗} && Z_I = U_I / I_I \\
&\text{输入导纳} && Y_I = I_I / U_I \\
&\text{电压传输函数} && T_U = U_O / U_I \\
&\text{电流传输函数} && T_I = I_O / I_I \\
&\text{传输阻抗} && Z_{TR} = U_O / I_I \\
&\text{传输导纳} && Y_{TR} = I_O / U_I
\end{aligned} \tag{6.2.2}$$

这些函数都是 S 的有理函数。

我们求解(6.2.1)式中的 X,并设输出 F 是解向量 X 的线性组合,即

$$F = e^T X$$

其中 e 为常数向量,则上式可写为

$$F = e^T T^{-1} B = e^T \frac{\hat{T}}{\det T} B \tag{6.2.3}$$

其中,$\hat{T}$ 为 T 的伴随矩阵,detT 为 T 的行列式。若(6.2.3)式中分子和分母无可约项,则所有的网络函数具有相同的分母形式,它等于系数矩阵 T 的行列式。

在复频域中,我们可以将电路方程表示为

$$T = G + SC \tag{6.2.4}$$

即用两个实矩阵 G 和 C 来描述方程系数矩阵。可以看出,T 的行列式和伴随矩阵 $\hat{T}$ 是 S 的多项式,而输出函数 F 是 S 的有理函数。可将 F 表示为以下两个等价的形式:

(1) 将 F 表示为多项式之比,即

$$F(S) = \frac{\sum_{i=0}^{n} a_i S^i}{\sum_{j=0}^{m} b_j S^j} \tag{6.2.5}$$

式中 a_i, b_j 为常数。

(2) 将 F 表示为(6.2.5)式分子和分母多项式根的形式,即

$$F(S) = K\frac{\prod_{i=0}^{n}(S - z_i)}{\prod_{j=0}^{m}(S - p_j)} \tag{6.2.6}$$

式中 z_i 和 p_j 分别是函数 $F(S)$ 的零点和极点。

如果我们令输入源为 1, 那么输出函数 F 即为电路的传输函数, 其形式可为

$$F(S) = \frac{N(S)}{D(S)} \tag{6.2.7}$$

其中 $N(S)$ 和 $D(S)$ 由(6.2.5)或(6.2.6)式定义。

2. 网络函数的计算机生成方法

在频域分析中, 我们在每个频率点上对电路方程 $TX = B$ 的系数矩阵 T 进行 LU 分解, 即对任何一个 S_i, 有

$$LUX = B \tag{6.2.8}$$

那么下三角阵 L 的对角元素之乘积就是 T 的行列式, 也就是网络函数的分母 $D(S)$, 于是

$$D(S_i) = \det T(S_i) = \det L(S_i) = \prod_{j=1}^{N} L_{jj} \tag{6.2.9}$$

式中 N 为电路方程的维数。

通过向前-向后代换求解(6.2.8)式中的 X, 并用 $F(S_i) = e^T X$ 表示网络输出, 则 F 是两个多项式之比, 即

$$F(S_i) = \frac{N(S_i)}{D(S_i)} = e^T X \tag{6.2.10}$$

(6.2.9)式已求出分母 $D(S_i)$ 值, 因此分子

$$N(S_i) = D(S_i)F(S_i) \tag{6.2.11}$$

由于采用改进节点法填入与频率有关的元件, 在公式中不出现 1/S 项, 这样就很容易地把 $F(S)$ 的分子 $N(S)$ 和分母 $D(S)$ 分离开来。

我们在频率取值范围内, 选取多个 S_i 值, 重复上述步骤, 便可得到 $\{S_i, N(S_i)\}$ 和 $\{S_i, D(S_i)\}$ 的点集。下一个问题是如何确定多项式 N 和 D 的系数。

3. 求解 $N(S)$ 和 $D(S)$ 的系数

我们采用单位圆上的多项式内插法求解 $N(S)$ 和 $D(S)$ 的系数。

假定有 $n+1$ 个不同的点 $\{x_i, y_i\}$, $y_i = f(x_i)$, $i = 0, 1, \cdots, n$; x_i 和 y_i 可以是实数也可以是复数。构造一个多项式 $P_n(x)$, 使它通过这 $n+1$ 个已知点, 即

$$P_n(x) = \sum_{j=0}^{n} a_j x^j \tag{6.2.12}$$

将 x_i 代入多项式(6.2.12), 便可得到含有未知数 a_j 的方程组

$$a_0 + a_1 x_i + a_2 x_i^2 + \cdots + a_n x_i^n = y_i \qquad i = 0, 1, \cdots, n \tag{6.2.13}$$

写成矩阵形式

$$\begin{matrix} 1 & x_0 & x_0^2 & \cdots & x_0^n \\ 1 & x_1 & x_1^2 & \cdots & x_1^n \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ 1 & x_n & x_n^2 & \cdots & x_n^n \end{matrix} \begin{matrix} a_0 \\ a_1 \\ \vdots \\ a_n \end{matrix} = \begin{matrix} y_0 \\ y_1 \\ \vdots \\ y_n \end{matrix} \tag{6.2.14}$$

或

$$Xa = Y \tag{6.2.15}$$

其中，$X = [x_i^j]$　　$i, j = 0, 1, \cdots, n$

x_i 的最佳选择是均匀分布在复平面内单位圆上的 x_i 点集。在单位圆上的点是

$$x_0 = 1$$

$$x_k = \exp\frac{2k\pi\sqrt{-1}}{n+1} \tag{6.2.16}$$

引入代换式

$$w = \exp\frac{2\pi\sqrt{-1}}{n+1} \tag{6.2.17}$$

则有

$$x_k = w^k \tag{6.2.18}$$

和

$$X = [w^{ij}] \tag{6.2.19}$$

可以证明

$$X^{-1} = \frac{1}{n+1}[w^{-ij}] = \frac{1}{n+1}X^* \tag{6.2.20}$$

式中 X^* 表示转置共轭矩阵。以此算出(6.2.19)和(6.2.20)式的乘积，即

$$XX^{-1} = \frac{1}{n+1}[w^{ij}][w^{-ij}] = 1 \tag{6.2.21}$$

这里 1 表示单位矩阵。上式的典型项的形式为

$$\frac{1}{n+1}\sum_{k=0}^{n} w^{ik}w^{-kj} = 1 \tag{6.2.22}$$

令 $i = j$，则

$$\frac{1}{n+1}\sum_{k=0}^{n} w^{ik}w^{-ik} = \frac{1}{n+1}\sum_{k=0}^{n} w^0 = 1 \tag{6.2.23}$$

对于 $i \neq j$ 的项，有

$$\frac{1}{n+1}\sum_{k=0}^{n} w^{ik}w^{-kj} = \frac{1}{n+1}\sum_{k=0}^{n} (w^{i-j})^k$$

上式右边的项是一个几何级数，应用级数求和法则，得

$$W^{i-j\ n+1} = \exp\frac{2\pi\sqrt{-1}}{n+1}(n+1)(i-j)$$

$$= \exp 2\pi\sqrt{-1}(i-j) = 1$$

故有

$$\frac{1}{n+1}\sum_{k=0}^{n} (w^{i-j})^k = \frac{1}{n+1}\frac{1-(w^{i-j})^{n+1}}{1-w^{i-j}} = 0 \tag{6.2.24}$$

实际上,(6.2.23)和(6.2.24)式是对(6.2.20)式的证明。对于(6.2.16)式所定义的点,方程(6.2.5)的解是

$$a = X^{-1}Y = \frac{1}{n+1}[w^{-ij}]Y$$

或

$$a_j = \frac{1}{n+1}\sum_{k=0}^{n} y_k w^{-jk} \tag{6.2.25}$$

在 x_k 点,原多项式(6.2.12)可以写成

$$y_k = \sum_{j=0}^{n} a_j w^{jk} \tag{6.2.26}$$

式(6.2.25)和(6.2.26)互为对方的解,这叫做离数傅立叶变换(DFT)。当 $n+1=2^m$(m 为正整数)时,又被称为快速傅立叶变换。

4. 求零极点

求解出网络函数的分子、分母多项式以后,下一步就是求解多项式的根,即网络函数的零点与极点。求多项式的根的算法很多,也较为成熟,比如分离降次法、牛顿-拉夫森迭代法、拉格尔(Laguerre)法等等。读者可以查阅有关书籍和文献,在此我们就不介绍了。

综上所述,由电路方程求解网络函数的零极点的算法,可归结为如下步骤:

(1) 建立电路方程 $TX=B$,不允许出现 $1/S$ 项,并置源值为 1。

(2) 应先估算网络函数的阶 n。最简单且最保险的方法是假定 n 等于电路中电感数与电容数的和。

(3) 在单位圆上选取等间隔点 S_i, $i=0,1,\cdots,n$。

(4) 设 $i=0$。

(5) 用 LU 分解算法求解电路方程 $T(S_i)X=B$。

(6) 计算 $T(S_i)$的行列式,即 $D(S_i)$值,有

$$D(S_i) = \prod_{k=1}^{L\text{的维数}} L_{kk}$$

并将 $D(S_i)$值存入 D_i 中。

(7) 根据选择的输出 F,计算 $N(S_i)=D_iF(S_i)$,并存入 N_i 中。

(8) 若 $i<n$,令 $i=i+1$,转步骤(5),否则转步骤(9)。

(9) 对点集$\{S_i, N_i\}$,用 DFT 生成分子多项式。

(10) 对点集$\{S_i, D_i\}$,用 DFF 生成分母多项式。

(11) 采用多项式求根算法,求解出分子多项式的根(即零点)以及分母多项的根(即极点)。

下面通过一个简单实例说明求解网络函数零极点的算法。

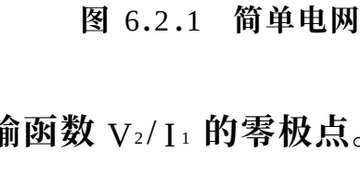


图 6.2.1 简单电网络

例 6.2.1 某一电网络如图 6.2.1 所示,求网络传输函数 V_2/I_1 的零极点。

用节点法列电网络方程为

$$\begin{bmatrix} 1+3S & -3S \\ -3S & 2+3S \end{bmatrix} \begin{bmatrix} V_1 \\ V_2 \end{bmatrix} = \begin{bmatrix} 1 \\ 0 \end{bmatrix}$$

电路中只有一个电容,故网络阶数 n= 1。我们在单位圆上选取两个均匀分布的点: $S_0 = 1$, $S_1 = -1$。将 $S_0 = 1$ 代入方程系数矩阵,再进行 LU 分解,得

$$\begin{bmatrix} 4 & 0 \\ -3 & 11/4 \end{bmatrix} \begin{bmatrix} 1 & -3/4 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} V_1(S_0) \\ V_2(S_0) \end{bmatrix} = \begin{bmatrix} 1 \\ 0 \end{bmatrix}$$

则行列式值 $D(S_0) = 4 \times \frac{11}{4} = 11$。利用向前-向后代换,求解出该方程的解为

$$\begin{bmatrix} V_1(S_0) \\ V_2(S_0) \end{bmatrix} = \begin{bmatrix} 5/11 \\ 3/11 \end{bmatrix}$$

即

$$V_2(S_0) = N(S_0)/D(S_0) = 3/11$$

那么,

$$N(S_0) = V_2(S_0)D(S_0) = (3/11) \times 11 = 3$$

将 $S_1 = -1$ 代入方程系数矩阵,同样进行 LU 分解,得

$$\begin{bmatrix} -2 & 0 \\ 3 & 7/2 \end{bmatrix} \begin{bmatrix} 1 & -3/2 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} V_1(S_1) \\ V_2(S_1) \end{bmatrix} = \begin{bmatrix} 1 \\ 0 \end{bmatrix}$$

分母 $D(S_1) = -2 \times (7/2) = -7$,方程解为

$$\begin{bmatrix} V_1(S_1) \\ V_2(S_1) \end{bmatrix} = \begin{bmatrix} 1/7 \\ 3/7 \end{bmatrix}$$

于是

$$N(S_1) = V_2(S_1)D(S_1) = (3/7) \times (-7) = -3$$

所以,确定分母内插点为(1, 11)和(-1, -7),由公式(6.2.25)算出

$$b_0 = \frac{1}{2}(D_0 + D_1) = 2$$

$$b_1 = \frac{1}{2}(D_0 - D_1) = 9$$

故

$$D(S) = 9S + 2$$

对于分子,相应内插点为(1, 3)和(-1, -3),故内插多项式为

$$N(S) = 3S$$

所以传输函数是

$$Z_{TR} = V_2/I_1 = \frac{3S}{9S+2}$$

这是一个很简单的情况,可以看出零点在 S= 0,极点在 S= -9/2。

在 Berkeley 的高于 SPICE3E 版本的程序中,以及许多商用电路仿真软件中都有零极点分析的功能,能对指定的传输函数计算其全部零极点值。

习　　题

6.1　用改进节点法列出题图 6.1 所示电路在角频率为 ω时电路的节点方程组，其中晶体管采用 π型 EM2模型，信号源 E_A 值为 1

6.2　请列出当角频率为 ω时，题图 6.2 所示各电路的电路方程。

6.3　请列出题图 6.3 所示电路在角频率 ω= 10 时的电路方程。并用 PSpice 程序求出输出电流 I_o。

6.4　请用 PSpice 程序计算习题 4.7 中电路的频响特性，并将该电路的通频带调到大于 8kHz。

6.5　请用 PSpice 程序计算习题 4.8 中电路的频响特性，以及当频率 f = 5kHz 时电路的输入、输出阻抗。

题图　6.1

题图　6.2

题图　6.3

6.6　请用 PSpice 程序计算题图 6.6 所示反馈放大器的增益和输入电阻(工作频率 f = 10kHz)。

题图　6.6

6.7　求题图 6.7 所示网络中,网络函数 U_2/I_1 的零极点。

题图　6.7

第 7 章　灵敏度分析

7.1　引　　言

前面几章中我们所涉及到的电路的元件参数值是其标称值,而实际电路中的元件值都不可避免地存在着误差。元件值的误差可能是在元件生产制造中所造成的,或者是由于温度变化、老化等环境条件变化引起的。元件参数值的误差必然会引起电路输出特性的误差,这些误差一般不能忽略,因此在设计电子电路时,研究电路中各元件参数的变化对电路特性的影响是很重要的问题。我们引入“灵敏度”的概念来表示这类变化关系的一种度量。

灵敏度(sensitivity)是指网络函数对网络元器件参数的敏感程度。设 T 为网络函数,它可以是节点电位、支路电流等等;p 为网络中的元器件参数,如电导、电容、温度、晶体管模型参数等等;则“相对灵敏度”定义为

$$S_p^T = \frac{T}{p}\frac{p}{T} = \frac{T/T}{p/p} = \frac{\ln T}{\ln p} \tag{7.1.1}$$

即网络函数 T 的相对变化量与网络参数 p 的相对变化量之比。此定义中假定变化量足够小。我们也称(7.1.1)式定义的灵敏度为“归一化灵敏度”。当 T 或 p 为零时,就应采用“非归一化灵敏度”,它是网络函数 T 对网络参数 p 的偏导数,即

$$S_p^T = \frac{T}{p} \tag{7.1.2}$$

灵敏度分析是计算灵敏度的方法。用直接对网络函数求导的方法计算灵敏度已很少用了,因为除了极为简单的网络或电路以外,大多数网络的网络函数都较为复杂,直接求导很困难。目前,电路 CAD 中常用的灵敏度计算方法有伴随网络法、导数网络法及符号网络函数法等等。这些方法把函数的求导问题转化为网络分析问题,避免了对复杂电路函数的求导。本章主要介绍伴随网络法和导数网络法。

我们要强调的是,通过灵敏度分析不仅能直接得知网络参数变化对网络特性的影响,而更重要的是,灵敏度分析是容差分析和电路优化设计的基础。

7.2　伴随网络法

伴随网络法是计算灵敏度的一个重要方法。它通过求解伴随网络方程,可得到网络的一个输出量对全部网络参数分别变化的灵敏度值。伴随网络法建立在特勒根(Tellegen)定理的基础上。下面介绍特勒根定理和伴随网络的概念,以及由此求解灵敏度的方法。

7.2.1 特勒根定理和伴随网络的构成

定理 设有两个网络 N 和 $\hat{N}$，它们具有相同的拓扑结构，且两网络对应的支路取相同的编号及正方向。假定它们的支路电压与支路电流分别用向量 U_b, I_b 和 $\hat{U}_b, \hat{I}_b$ 表示，则有特勒根关系式

$$U_b^T \hat{I}_b = I_b^T \hat{U}_b = \hat{U}_b^T I_b = \hat{I}_b^T U_b = 0 \tag{7.2.1}$$

证明 由于网络 N 和 $\hat{N}$ 结构相同，因此它们的关联矩阵相同，即 $A = \hat{A}$。由 KCL 和 KVL 定律，有

$$AI_b = 0 \tag{7.2.2}$$

$$U_b = A^T V_n \tag{7.2.3}$$

将(7.2.2)和(7.2.3)代入特勒根关系式(7.2.1)，则有

$$U_b^T \hat{I}_b = (A^T V_n)^T \hat{I}_b = V_n^T (A \hat{I}_b) = 0$$

同理可证明 $I_b^T \hat{U}_b = 0$，$\hat{U}_b^T I_b = 0$ 和 $\hat{I}_b^T U_b = 0$，即特勒根关系式(7.2.1)成立。

特勒根定理是电网络的能量守恒定律，只要两个网络 N 和 $\hat{N}$ 具有相同的拓扑结构，并不一定要求有相同的支路元件，而且即使它们的支路电压和支路电流是在不同时刻测定的，特勒根定理仍然成立。

推论 设网络 N 和网络 $\hat{N}$ 满足特勒根定理的条件，即具有相同的拓扑结构。设两网络的支路电压和支路电流向量分别是 U_b, I_b 和 $\hat{U}_b, \hat{I}_b$。假定网络 N 由于某个元件参数发生微小变化，其支路电压和支路电流向量变为 $U_b + dU_b$ 和 $I_b + dI_b$，则特勒根关系式仍然成立。即

$$\begin{aligned} \hat{I}_b^T (U_b + dU_b) &= 0 \\ \hat{U}_b^T (I_b + dI_b) &= 0 \end{aligned} \tag{7.2.4}$$

根据特勒根定理，将 $\hat{I}_b^T U_b = 0$ 和 $\hat{U}_b^T I_b = 0$ 代入(7.2.4)式，得

$$\hat{I}_b^T dU_b = 0 \tag{7.2.5}$$

$$\hat{U}_b^T dI_b = 0 \tag{7.2.6}$$

或写为

$$\hat{I}_b^T dU_b - \hat{U}_b^T dI_b = 0 \tag{7.2.7}$$

此式是伴随网络法计算灵敏度的基本关系式。

为了利用特勒根定理计算网络灵敏度，需要构造伴随网络。我们定义一个网络 $\hat{N}$，若它与原网络 N 满足下列条件，则称网络 $\hat{N}$ 为原网络 N 的伴随网络。

(1) 两个网络具有相同的拓扑结构，即关联矩阵相等：$\hat{A} = A$。

(2) 两个网络中,除独立源外,它们的支路阻抗矩阵(或支路导纳矩阵)互为转置,即

$$\hat{Z}_b = Z_b^T \quad 或 \quad \hat{Y}_b = Y_b^T$$

(3) 两个网络的独立源具有相同的性质,但不一定具有相同的数值。

一个网络中的独立电压源和独立电流源,在另一个网络中对应为短路支路(零值电压源)和开路支路(零值电流源)。我们用表7.2.1说明网络 $\hat{N}$ 与N中独立源的对应关系。在原网络中的电压输出支路和电流输出支路,在伴随网络 $\hat{N}$ 中对应为单位电流源和单位电压源。

表 7.2.1　伴随网络与原网络中独立源对应关系

7.2.2　线性网络的伴随网络法

由特勒根定理推导出的公式(7.2.7),也可以表示为

$$\sum_{k=1}^{m} (\hat{I}_k dU_k - \hat{U}_k dI_k) = 0 \tag{7.2.8}$$

式中 m 为网络中的支路总数。该式反映了具有相同拓扑结构的网络 N 与伴随网络 $\hat{N}$ 之间的关系。我们利用这个关系式推导线性网络灵敏度的计算公式,称之为线性网络灵敏度的伴随网络法。

我们通过一个实例介绍应用伴随网络法推导线性网络中元器件参数的灵敏度公式。

例 7.2.1　某电网络 N 如图 7.2.1 所示,用伴随网络法推导网络中输出电压 U_o 对

电阻 R_1、电导 G_2、恒压源 E 和恒流源 I_S 的灵敏度。

图 7.2.1　原网络 N

图 7.2.2　伴随网络 $\hat{N}$

解　按照构造伴随网络的原则，建立图 7.2.1 所示电网络的伴随网络 $\hat{N}$，如图 7.2.2 所示。原网络中的恒压源，在伴随网络中对应为一短路支路；原网络中的恒流源，在伴随网络中由开路支路代替；原网络中的电阻或电导支路，在伴随网络中仍为电阻或电导支路；输出电压支路在 $\hat{N}$ 中用单位电流源取代。

1．求 U_O 对电阻的灵敏度

设 R_1 是网络 N 中产生微小变化的参数，其值为 $R_1+\mathrm{d}R_1$。根据(7.2.8)，此网络 N 与伴随网络 $\hat{N}$ 的相应关系为

$$\sum_{k=1}^{5}(\hat{I}_k\mathrm{d}U_k-\hat{U}_k\mathrm{d}I_k)=0 \tag{7.2.9}$$

即

$$(\hat{I}_E\mathrm{d}E-\hat{E}\,\mathrm{d}I_E)+(\hat{I}_S\mathrm{d}U_S-\hat{U}_S\mathrm{d}I_S)+(\hat{I}_{R_1}\mathrm{d}U_{R_1}-\hat{U}_{R_1}\mathrm{d}I_{R_1})$$

$$+(\hat{I}_{G_2}\mathrm{d}U_{G_2}-\hat{U}_{G_2}\mathrm{d}I_{G_2})+(\hat{I}_O\mathrm{d}U_O-\hat{U}_O\mathrm{d}I_O)=0 \tag{7.2.10}$$

(1) 恒压源支路：由于在原网络 N 中，恒压源电压不随 R_1 变化而改变，$\mathrm{d}E=0$；在伴随网络 $\hat{N}$ 中，对应支路为短路支路，$\hat{E}=0$；故(7.2.10)式的第一个括弧内两项均为零。

(2) 恒流源支路：原网络中，恒流源电流不随 R_1 变化而改变，即 $\mathrm{d}I_S=0$；在伴随网络中对应支路为开路，$\hat{I}_S=0$；故式(7.2.10)中的第二个括弧内的两项也等于零。

(3) 导纳 G_2 支路：原网络中，由于 R_1 变化而变化的 $\mathrm{d}I_{G_2}=\mathrm{d}U_{G_2}G_2$；而且在伴随网络中的 $\hat{G}_2=G_2$，故第 4 个括弧内两项仍为零，即

$$\hat{I}_{G_2}\mathrm{d}U_{G_2}-\hat{U}_{G_2}\mathrm{d}I_{G_2}=\hat{U}_{G_2}\hat{G}_2\,\mathrm{d}U_{G_2}-\hat{U}_{G_2}\mathrm{d}U_{G_2}G_2=0$$

(4) 电阻 R_1 支路：这是参数产生变化的支路，根据欧姆定律，有

$$U_{R_1}+\mathrm{d}U_{R_1}=(I_{R_1}+\mathrm{d}I_{R_1})(R_1+\mathrm{d}R_1)$$

$$=I_{R_1}R_1+\mathrm{d}I_{R_1}R_1+I_{R_1}\mathrm{d}R_1+\mathrm{d}I_{R_1}\mathrm{d}R_1$$

令 $\mathrm{d}I_{R_1}\mathrm{d}R_1\approx 0$，则有

$$\mathrm{d}U_{R_1}\approx\mathrm{d}I_{R_1}R_1+I_{R_1}\mathrm{d}R_1$$

将上式代入(7.2.10)式中的第三个括弧,得

$$\hat{I}_{R_1}\mathrm{d}U_{R_1} - \hat{U}_{R_1}\mathrm{d}I_{R_1} = \hat{I}_{R_1}\mathrm{d}I_{R_1}R_1 + \hat{I}_{R_1}I_{R_1}\mathrm{d}R_1 - \hat{I}_{R_1}\hat{R}_1\mathrm{d}I_{R_1}$$

由于 $\hat{R}_1 = R_1$,第 3 个括弧中最后化简为只剩一项: $I_{R_1}\hat{I}_{R_1}\mathrm{d}R_1$。

(5) 输出支路: 原网络输出支路为 $I_O = 0$; 伴随网络对应输出支路为单位电流源,即 $\hat{I}_O = 1$; 故第 5 个括弧内为

$$\hat{I}_O\mathrm{d}U_O - \hat{U}_O\mathrm{d}I_O = \mathrm{d}U_O$$

通过上面推导可以看出,只有产生扰动的 R_1 支路和输出支路对应的项不为零,于是式(7.2.10)化简为

$$\hat{I}_{R_1}I_{R_1}\mathrm{d}R_1 + \mathrm{d}U_O = 0$$

由此得到

$$\frac{\mathrm{d}U_O}{\mathrm{d}R_1} = -I_{R_1}\hat{I}_{R_1} \tag{7.2.11}$$

或写为

$$S_{R_1}^{U_O} = \frac{U_O}{R_1} = -I_{R_1}\hat{I}_{R_1} \tag{7.2.12}$$

这就是输出电压 U_O 对电阻 R_1 的灵敏度公式。它在数值上等于原网络 N 中流过 R_1 支路的电流 I_{R_1} 与伴随网络 $\hat{N}$ 中流过相应的 $\hat{R}_1$ 支路的电流 $\hat{I}_{R_1}$ 的乘积。量纲是 V/Ω。

2. 求 U_O 对电导的灵敏度

通过前面对电阻灵敏度公式的推导可以看出,凡是在网络 N 中没有产生扰动的参数支路所对应的(7.2.10)式中的各项均为零; 只有产生扰动的参数支路和输出支路对应的项不为零。因此,当网络 N 中电导 G_2 产生扰动时,在式(7.2.10)中相应的不为零项为

$$(\hat{I}_{G_2}\mathrm{d}U_{G_2} - \hat{U}_{G_2}\mathrm{d}I_{G_2}) + \mathrm{d}U_O = 0 \tag{7.2.13}$$

由于 G_2 产生扰动,根据欧姆定律有

$$\begin{aligned} I_{G_2} + \mathrm{d}I_{G_2} &= (U_{G_2} + \mathrm{d}U_{G_2}) + (G_2 + \mathrm{d}G_2) \\ &= U_{G_2}G_2 + \mathrm{d}U_{G_2}G_2 + U_{G_2}\mathrm{d}G_2 + \mathrm{d}U_{G_2}\mathrm{d}G_2 \end{aligned}$$

令 $\mathrm{d}U_{G_2}\mathrm{d}G_2 = 0$,则有

$$\mathrm{d}I_{G_2} \approx \mathrm{d}U_{G_2}G_2 + U_{G_2}\mathrm{d}G_2$$

代入(7.2.13)式,由于 $\hat{G}_2 = G_2$,故有

$$\hat{U}_{G_2}\hat{G}_2\,\mathrm{d}U_{G_2} - \hat{U}_{G_2}\mathrm{d}U_{G_2}G_2 - \hat{U}_{G_2}U_{G_2}\mathrm{d}G_2 + \mathrm{d}U_O = 0$$

化简为

$$-\hat{U}_{G_2}U_{G_2}\mathrm{d}G_2 + \mathrm{d}U_O = 0$$

即

$$\frac{\mathrm{d}U_O}{\mathrm{d}G_2} = U_{G_2}\hat{U}_{G_2} \tag{7.2.14}$$

或写为

$$S_{G_2}^{U_O} = \frac{U_O}{G_2} = U_{G_2}\hat{U}_{G_2} \tag{7.2.15}$$

这就是输出变量 U_O 对电导 G_2 的灵敏度公式。它在数值上等于原网络 N 中电导 G_2 两端电压 U_{G_2} 与伴随网络 $\hat{N}$ 中相应 $\hat{G}_2$ 支路电压 $\hat{U}_{G_2}$ 的乘积，量纲是 V/S。

3. 求 U_O 对独立源的灵敏度

(1) 如果独立电压源的电压产生扰动，在式(7.2.10)中相应的不为零项为

$$(\hat{I}_E dE - \hat{E}\, dI_E) + dU_O = 0 \tag{7.2.16}$$

由于在 $\hat{N}$ 中，恒压源支路为短路支路，有 $\hat{E} = 0$，故(7.2.16)变为

$$\hat{I}_E dE + dU_O = 0$$

即

$$\frac{dU_O}{dE} = -\hat{I}_E \tag{7.2.17}$$

或写成

$$S_E^{U_O} = \frac{U_O}{E} = -\hat{I}_E \tag{7.2.18}$$

这就是输出电压 U_O 对独立电压源 E 的灵敏度公式，它在数值上等于伴随网络中负的恒压源支路电流，量纲是 V/V。

(2) 如果独立电流源的电流产生扰动，在式(7.2.10)中相应的不为零项为

$$(\hat{I}_S dU_S - \hat{U}_S dI_S) + dU_O = 0 \tag{7.2.19}$$

由于在 $\hat{N}$ 中，电流源支路为开路支路，有 $\hat{I}_S = 0$，故(7.2.19)式变为

$$-\hat{U}_S dI_S + dU_O = 0$$

即

$$\frac{dU_O}{dI_S} = \hat{U}_S \tag{7.2.20}$$

或写为

$$S_{I_S}^{U_O} = \frac{U_O}{I_S} = \hat{U}_S \tag{7.2.21}$$

这就是输出电压 U_O 对独立电流源的灵敏度公式，它在数值上等于伴随网络中独立电流源支路的电压，量纲是 V/A。

通过例 7.2.1 我们可以看出，用伴随网络法推导出的元件参数的灵敏度公式常常由两部分组成，一部分是该元件在原网络中的支路电流或电压，另一部分是该元件在伴随网络中的支路电流或电压。这两部分的乘积构成了该元件的灵敏度公式。因此，我们归纳出伴随网络法的灵敏度公式有如下特点：

1) 对阻抗类元件，灵敏度公式表示为原网络中其支路电流乘以伴随网络中其支路电流，再加一负号。例如前面推导的电阻灵敏度公式

$$\frac{U_O}{R} = -I_R\hat{I}_R \tag{7.2.22}$$

对电感元件，其灵敏度公式也可按此方式推导，即有

$$\frac{U_O}{(j\omega L)} = -\ I_L \hat{I}_L$$

即

$$\frac{U_O}{L} = -\ j\omega I_L \hat{I}_L \tag{7.2.23}$$

2) 对导纳类元件，灵敏度公式表示为原网络中导纳支路电压乘以伴随网络中其支路电压。例如前面推导的电导灵敏度公式

$$\frac{U_O}{G} = U_G \hat{U}_G \tag{7.2.24}$$

对电容，其灵敏度公式可按此推导，即有

$$\frac{U_O}{(j\omega C)} = U_C \hat{U}_C$$

即

$$\frac{U_O}{C} = j\omega U_C \hat{U}_C \tag{7.2.25}$$

3) 对独立源器件，灵敏度公式仅由伴随网络中其对应的支路电流或电压来描述。对独立电压源，灵敏度公式表示为伴随网络中其负的支路电流，对独立电流源，灵敏度公式表示为伴随网络中其支路电压，于是有

$$\frac{U_O}{E} = -\ \hat{I}_E \tag{7.2.26}$$

$$\frac{U_O}{I_S} = \hat{U}_S \tag{7.2.27}$$

4) 对受控源器件，灵敏度公式由原网络中控制支路的电压或电流与伴随网络中被控支路的电流或电压的乘积构成。如果是压控源，取原网络的支路电压，如果是流控源，取原网络的支路电流；如果被控的是电压源，取伴随网络中其支路电流，如果被控的是电流源，取伴随网络中其支路电压。例如，电压控制电流源(VCCS)，其灵敏度公式由原网络中控制电压 U_1 与伴随网络中被控支路电压 $\hat{U}_2$ 的乘积组成，即

$$\frac{U_O}{g_m} = U_1 \hat{U}_2 \tag{7.2.28}$$

电流控制电流源(CCCS)的灵敏度公式由原网络中控制电流 I_1 与伴随网格中被控支路电压 $\hat{U}_2$ 的乘积组成，即

$$\frac{U_O}{\beta} = I_1 \hat{U}_2 \tag{7.2.29}$$

表 7.2.2 中列出了常用的各种线性元器件采用伴随网络法的灵敏度公式。

7.2.3 非线性网络的伴随网络法

如果网络 N 是非线性网络，即含有非线性元件，如二极管和晶体管等，这些非线性元件支路一般可以等效为电压控制的非线性电导或电流控制的非线性电阻。下面推导采用伴随网络法的非线性电导的灵敏度公式。

网络 N 中含有非线性元件，其特性方程是

表 7.2.2　线性元件的伴随网络灵敏度公式

$$I_g = f(U_g, p) \tag{7.2.30}$$

即为电压控制的非线性电导，式中 I_g，U_g 分别为非线性元件的支路电流和支路电压，p 为参量。如果在网络 N 中，该非线元件的某一参量 p 产生扰动，则它和输出支路在(7.2.10)

式中对应的不为零项是

$$(\hat{I}_g dU_g - \hat{U}_g dI_g) + dU_O = 0 \tag{7.2.31}$$

(1) 如果参数 p 没有发生变化,则有

$$\hat{I}_g dU_g - \hat{U}_g dI_g = 0 \tag{7.2.32}$$

由于

$$dI_g = \frac{f}{U_g} dU_g + \frac{f}{p} dp = \frac{f}{U_g} dU_g$$

代入(7.2.32),得

$$\hat{I}_g dU_g - \hat{U}_g \frac{f}{U_g} dU_g = 0$$

即

$$\hat{I}_g = \hat{U}_g \frac{f}{U_g} \tag{7.2.33}$$

这说明在伴随网络 $\hat{N}$ 中,非线性支路可用一个工作点上的线性电导 $\frac{f}{U_g}$ 来代替,如图 7.2.3。也就是说,如果原网络 N 是个非线性网络,则其伴随网络 $\hat{N}$ 是个线性网络。

图 7.2.3 非线性电导的伴随网络模型

(2) 当参量 p 发生变化时,有

$$dI_g = \frac{f}{U_g} dU_g + \frac{f}{p} dp$$

代入(7.2.31)式,得

$$\hat{I}_g dU_g - \hat{U}_g \frac{f}{U_g} dU_g - \hat{U}_g \frac{f}{p} dp + dU_O = 0$$

将(7.2.33)式代入上式,得

$$-\hat{U}_g \frac{f}{p} dp + dU_O = 0$$

即

$$\frac{dU_O}{dp} = \hat{U}_g \frac{f}{p} \tag{7.2.34}$$

这就是非线性电导的灵敏度公式,也可写为

$$S_p^{U_O} = \frac{U_O}{p} = \hat{U}_g \frac{f}{p} \tag{7.2.35}$$

例 7.2.2 二极管为非线性器件，其特性方程为 $I_D = I_S(e^{U_D/V_T} - 1)$，求二极管饱和电流 I_S 的灵敏度公式。

解 按照式(7.2.35)，网络输出变量 U_O 对二极管参数 I_S 的灵敏度公式为

$$S_{I_S}^{U_O} = \frac{U_O}{I_S} = \hat{U}_D \frac{I_D}{I_S} = \hat{U}_D(e^{U_D/V_T} - 1) \tag{7.2.36}$$

如果网络中非线性元件的特性方程用电流控制的非线性电阻形式描述，即

$$U_R = f(I_R, p) \tag{7.2.37}$$

则非线性电阻的灵敏度公式为

$$S_p^{U_O} = \frac{U_O}{p} = -\hat{I}_R \frac{f}{p} \tag{7.2.38}$$

7.2.4 伴随网络方程的建立及其求解

设原网络方程为

$$TX = B \tag{7.2.39}$$

伴随网络方程为

$$\hat{T}\hat{X} = \hat{B} \tag{7.2.40}$$

前面曾简述过由原网络 N 建立伴随网络 $\hat{N}$ 的三条原则：

(1) 网络 $\hat{N}$ 与网络 N 有相同的拓扑结构。

(2) 网络 $\hat{N}$ 与网络 N 的导纳(或阻抗)矩阵互为转置。

(3) 两网络的独立源性质相同，数值不同。

由这些原则可以看出，只要我们令 $\hat{T} = T^T$，就能够满足前两条原则，即伴随网络方程的系数矩阵是原网络方程系数矩阵的转置。第(3)条原则是描述独立源。独立源对方程的贡献反映在方程的右端向量。在伴随网络中，独立电压源短路，独立电流源开路，故它们对伴随网络方程的右端向量 $\hat{B}$ 没有贡献。所以 $\hat{B}$ 中只含有输出支路的单位电流源或单位电压源所对应的 ± 1 项。如果输出支路有一端接地，则 $\hat{B}$ 中在输出端对应位置只有一个 -1 项，其它项均为零。这样，我们可将伴随网络方程写为如下形式：

$$T^T\hat{X} = \hat{B} \tag{7.2.41}$$

综上所述，伴随网络方程不必由建立伴随网络而形成，它可以直接由原网络方程的转置而形成。故也称伴随网络法为转置系统法。

求解伴随网络方程(7.2.41)，可以直接利用原网络方程系数矩阵的 LU 分解结果

$$T = LU$$

则

$$T^T = U^T L^T$$

将上式代入(7.2.41)式，有

$$U^{T}L^{T}\hat{X} = \hat{B}$$

令 $$L^{T}\hat{X} = \hat{y}$$

则 $$U^{T}\hat{Y} = \hat{B}$$

由于利用了原网络 LU 分解的结果，求解伴随网络方程的乘除运算量仅是向前、向后替代所需的乘除次数，它比求解原网络方程的运算量少得多。

现将伴随网络法求稳态灵敏度的步骤归纳如下：

(1) 求解原网络方程 $TX = B$，得到原网络各支路电压和支路电流信息。

(2) 建立伴随网络方程 $T^{T}\hat{X} = \hat{B}$。其系数矩阵是原网络系数矩阵的转置，如果是非线性网络，则应是非线性迭代收敛后的原网络系数矩阵的转置。右端向量 $\hat{B}$ 中只需填入输出支路的贡献，它是一个最多含有两个非零元的向量。

(3) 求解伴随网络方程，得到伴随网络中各支路电压和支路电流信息。

(4) 根据原网络和伴随网络方程的结果，利用表 7.2.2 所列各元件灵敏度公式，计算出输出变量 U_o（或 I_o）对网络中所有元件参数的灵敏度值。

(5) 如果还进一步求网络中另外一个输出变量对元件参数的灵敏度，则需重新填写伴随网络方程的右端向量 $\hat{B}$，然后重复第(3)、(4)步骤。

采用伴随网络法每求解一次伴随网络方程，只能计算出网络的一个输出变量对所有网络元件参数的灵敏度，如果还想计算其它输出变量的灵敏度，则需再次求解伴随网络方程。一般人们只对网络中少数几个输出变量的灵敏度感兴趣，故求解伴随网络的次数不会很多。但当网络较大时，每次所需计算的网络参数灵敏度值会很多。

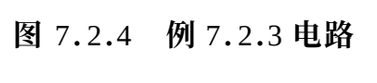


图 7.2.4　例 7.2.3 电路

例 7.2.3　用伴随网络法求图 7.2.4 电路中输出电压 U_o 对所有电路参数的灵敏度值。

解　原网络方程 $TX = B$ 为

$$\begin{bmatrix} \frac{1}{100} & -\frac{1}{100} & 0 & 1 & 0 \\ -\frac{1}{100} & \frac{1}{100} & 0 & 0 & 1 \\ 0 & 0 & \frac{1}{10} & 0 & 40 \\ 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \end{bmatrix} \begin{matrix} V_1 \\ V_2 \\ V_3 \\ I_E \\ I_2 \end{matrix} = \begin{matrix} 0 \\ 0 \\ 0 \\ 2 \\ 0 \end{matrix}$$

求解原网络方程，得到

$$X = [V_1 \quad V_2 \quad V_3 \quad I_E \quad I_2]^{T} = [2 \quad 0 \quad -8 \quad -0.02 \quad 0.02]^{T}$$

伴随网络方程为 $T^{T}\hat{X}=\hat{B}$，即

$$\begin{bmatrix} \frac{1}{100} & -\frac{1}{100} & 0 & 1 & 0 \\ -\frac{1}{100} & \frac{1}{100} & 0 & 0 & 1 \\ 0 & 0 & \frac{1}{10} & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 40 & 0 & 0 \end{bmatrix} \begin{bmatrix} \hat{V}_1 \\ \hat{V}_2 \\ \hat{V}_3 \\ \hat{I}_E \\ \hat{I}_2 \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ -1 \\ 0 \\ 0 \end{bmatrix}$$

解此伴随网络方程，得

$$\hat{X}=[\hat{V}_1\ \hat{V}_2\ \hat{V}_3\ \hat{I}_E\ \hat{I}_2]^{T}=[0\ \ 400\ \ -10\ \ 4\ \ -4]^{T}$$

由表 7.2.2 查得有关的灵敏度公式，并将相应 X，$\hat{X}$ 值代入，可分别得

$$S_E^{U_O}=\frac{U_O}{E}=-\hat{I}_E=-4\text{V/V}$$

$$S_{R_1}^{U_O}=\frac{U_O}{R_1}=-I_{R_1}\hat{I}_{R_1}=-\frac{U_{R_1}\hat{U}_{R_1}}{R_1^2}=-\frac{2\times(-400)}{100^2}=0.08\text{V/}\Omega$$

$$S_{R_2}^{U_O}=\frac{U_O}{R_2}=-I_{R_2}\hat{I}_{R_2}=-\frac{U_{R_2}\hat{U}_{R_2}}{R_2^2}=-\frac{(-8)\times(-10)}{10^2}=-0.8\text{V/}\Omega$$

$$S_{\beta}^{U_O}=\frac{U_O}{\beta}=I_2\hat{V}_3=0.02\times(-10)=-0.2\text{V/单位}\ \beta\text{值}$$

例 7.2.4　用伴随网络法求图 7.2.5 所示电路中直流输出电压 U_O 对二极管饱和电流 I_S 的灵敏度 $\frac{U_O}{I_S}$。已知二极管支路特性方程为 $I_D=I_S(e^{U_D/V_T}-1)$，其中 $I_S=8\times10^{-13}\text{A}$，$V_T=0.026\text{V}$。

图 7.2.5　例 7.2.4 电路

图 7.2.6　例 7.2.4 电路的伴随网络

解　原网络方程 $TX=B$ 为

$$\begin{bmatrix} \frac{1}{R} & -\frac{1}{R} & 1 \\ -\frac{1}{R} & \frac{1}{R}+G_{DM}^{k} & 0 \\ 1 & 0 & 0 \end{bmatrix} \begin{bmatrix} V_1 \\ V_2 \\ I_E \end{bmatrix} = \begin{bmatrix} 0 \\ -I_{DM}^{k} \\ E \end{bmatrix}$$

其中 G_{DM}^{k}, I_{DM}^{k}为采用 N-R 方法线性化时二极管的直流伴随模型参数。$G_{DM}^{k}=\frac{I_D}{U_D}=\frac{I_S}{V_T}\times e^{U_D^k/V_T}$, $I_{DM}^{k}=I_S(e^{U_D^k/V_T}-1)-G_{DM}^{k}U_D^k$。迭代收敛后,此方程解为

$$X=[V_1\quad V_2\quad I_E]^T=[2.00\quad 0.6098\quad -1.39\times 10^{-2}]^T$$

在伴随网络中,二极管用一电导$\frac{I_D}{U_D}$来代替,实际上就是原网络方程求解出的工作点时的电导 G_{DM}值。伴随网络如图 7.2.6 所示。

我们用原网络方程直流迭代收敛后的系数矩阵的转置,构成伴随网络方程,即

$$\begin{bmatrix}\frac{1}{R} & -\frac{1}{R} & 1\\ -\frac{1}{R} & \frac{1}{R}+G_{DM} & 0\\ 1 & 0 & 0\end{bmatrix}\begin{bmatrix}\hat{V}_1\\ \hat{V}_2\\ \hat{I}_E\end{bmatrix}=\begin{bmatrix}0\\ -1\\ 0\end{bmatrix}$$

其中

$$G_{DM}=\left.\frac{I_D}{U_D}\right|_{\text{工作点}}=\frac{I_S}{V_T}e^{V_2/V_T}=0.4721$$

解此伴随网络方程得

$$\hat{X}=[\hat{V}_1\quad \hat{V}_2\quad \hat{I}_E]^T=[0\quad -2.074\quad -2.074\times 10^{-2}]^T$$

由式(7.2.36),得到

$$S_{I_S}^{U_O}=\frac{U_O}{I_S}=\hat{U}_D\frac{I_D}{I_S}=\hat{V}_2(e^{V_2/V_T}-1)=-3.18\times 10^{10}\text{V/A}$$

7.3 网络灵敏度的应用

7.3.1 寄生参数的灵敏度

在实际电网络中往往存在有寄生参数,尤其是当电路工作在较高频率条件下时,寄生参数对电路性能的影响是不能忽略的,因此讨论寄生参数的灵敏度对电路分析有实际意义。

假定在理想电路中寄生参数的标称值是零,它对电路及电路方程本身不产生任何影响,但它的微小变化,有可能对电路性能产生很大影响。我们利用伴随网络法可以计算出寄生参数变化时,电路输出变量对寄生参数的灵敏度值。也就是说,在原网络中一个零值的寄生参数可能有非零的灵敏度值。

例 7.3.1　在图 7.3.1 所示的电网络中含有三个寄生元件,其值为 $C_1=0$, $G_2=0$ 和 $C_3=0$,用伴随网络法求该网络输出量 U_O 对这三个寄生元件参数的灵敏度值$\frac{U_O}{C_1}$, $\frac{U_O}{G_2}$和$\frac{U_O}{C_3}$。

图 7.3.1　例 7.3.1 电网络

解　网络方程如下:

$$\begin{bmatrix} G_1 + G_2 + SC_1 + SC_2 & -G_2 - SC_2 \\ -G_2 - SC_2 & G_2 + G_3 + SC_2 + SC_3 \end{bmatrix} \begin{bmatrix} V_1 \\ V_2 \end{bmatrix} = \begin{bmatrix} 1 \\ 0 \end{bmatrix}$$

式中 $S = j\omega = j$(由于 $\omega = 1$)。因 $C_1 = 0, G_2 = 0, C_3 = 0$, 故上式为

$$\begin{bmatrix} G_1 + SC_2 & -SC_2 \\ -SC_2 & G_3 + SC_2 \end{bmatrix} \begin{bmatrix} V_1 \\ V_2 \end{bmatrix} = \begin{bmatrix} 1 \\ 0 \end{bmatrix}$$

即

$$\begin{bmatrix} 1 + j & -j \\ -j & 1 + j \end{bmatrix} \begin{bmatrix} V_1 \\ V_2 \end{bmatrix} = \begin{bmatrix} 1 \\ 0 \end{bmatrix}$$

解此方程组, 得到

$$X = [V_1 \quad V_2]^T = \left[\frac{3 - j}{5} \quad \frac{2 + j}{5}\right]^T$$

伴随网络方程为

$$\begin{bmatrix} 1 + j & -j \\ -j & 1 + j \end{bmatrix} \begin{bmatrix} \hat{V}_1 \\ \hat{V}_2 \end{bmatrix} = \begin{bmatrix} 0 \\ -1 \end{bmatrix}$$

其解为

$$\hat{X} = [\hat{V}_1 \quad \hat{V}_2]^T = \left[\frac{-(2 + j)}{5} \quad \frac{-3 + j}{5}\right]^T$$

由 $X, \hat{X}$ 计算寄生参数 C_1, G_2 和 C_3 的灵敏度, 得

$$\frac{V_2}{C_1} = SV_1\hat{V}_1 = j\frac{(3 - j)}{5}\frac{(-2 - j)}{5} = \frac{1 - 7j}{25}$$

$$\frac{V_2}{G_2} = U_{G_2}\hat{U}_{G_2} = (V_1 - V_2)(\hat{V}_1 - \hat{V}_2) = \frac{1 - 2j}{5}\frac{1 - 2j}{5} = \frac{-3 - 4j}{25}$$

$$\frac{V_2}{C_3} = SV_2\hat{V}_2 = j\frac{2 + j}{5}\frac{-3 + j}{5} = \frac{1 - 7j}{25}$$

现将寄生参数灵敏度计算方法归纳如下:

(1) 假定寄生参数存在, 将其导纳 $G = 0$ 填入方程系数矩阵 T。当然它对方程系数矩阵无影响, 即它不改变原网络方程。

(2) 求解原网络方程 $TX = B$, 求得解 X。

(3) 寄生参数对伴随网络无影响, 也不改变伴随网络方程。求解伴随网络方程 $T^T\hat{X} = \hat{B}$, 求得 $\hat{X}$。

(4) 根据寄生参数的性质以及在网络的位置, 利用表 7.2.2 中的相应灵敏度公式, 由 $X, \hat{X}$ 计算各寄生参数的灵敏度值。

7.3.2　对于频率的灵敏度

在频域中, 电路方程 $TX = B$ 是个复数方程, 我们将其系数矩阵分为实系数矩阵和复系数矩阵两部分, 即

$$T = T_R + ST_I \tag{7.3.1}$$

式中 $S = j\omega$。

在频域中，电路方程的右端向量(即源向量)与频率 ω 无关，故

$$\frac{B}{\omega} = 0 \tag{7.3.2}$$

方程的系数矩阵中，与频率有关的项都在复系数子阵中，故有

$$\frac{T}{\omega} = jT_I \tag{7.3.3}$$

这就说明，网络中输出变量对频率的灵敏度仅仅和网络的电抗元件有关。因此，我们可以通过对电抗元件的灵敏度的计算来求解对频率的灵敏度。从表 7.2.2 中可查出，电容和电感元件的灵敏度公式为

$$\frac{U_O}{C} = j\omega U_C \hat{U}_C$$

$$\frac{U_O}{L} = -j\omega I_L \hat{I}_L$$

则对频率的灵敏度公式为

$$\frac{U_O}{\omega} = \frac{1}{\omega}\left(\sum_i C_i \frac{U_O}{C_i} + \sum_j L_j \frac{U_O}{L_j}\right) \tag{7.3.4}$$

即

$$\frac{U_O}{\omega} = \frac{1}{\omega}\left(\sum_i C_i\, j\omega U_{C_i}\hat{U}_{C_i} + \sum_j L_j\, -j\omega I_{L_j}\hat{I}_{L_j}\right) \tag{7.3.5}$$

其中 $U_O, U_{C_i}, \hat{U}_{C_i}, I_{L_j}, \hat{I}_{L_i}$ 通常为复数。输出变量 U_O 可写为

$$U_O = |U_O| e^{j\phi}$$

两边取对数

$$\ln U_O = \ln|U_O| + j\phi$$

将上式对参数 ω 求导，有

$$\frac{1}{U_O}\frac{U_O}{\omega} = \frac{1}{|U_O|}\frac{|U_O|}{\omega} + j\frac{\phi}{\omega}$$

或写为

$$\frac{|U_O|}{\omega} = |U_O| \mathrm{Re}\left(\frac{1}{U_O}\frac{U_O}{\omega}\right) \tag{7.3.6}$$

$$\frac{\phi}{\omega} = \mathrm{Im}\left(\frac{1}{U_O}\frac{U_O}{\omega}\right) \tag{7.3.7}$$

我们知道，群延时的定义为

$$\tau = -\frac{\phi}{\omega} = -\mathrm{Im}\left(\frac{1}{U_O}\frac{U_O}{\omega}\right) \tag{7.3.8}$$

即能用对频率的灵敏度来表示网络的群延时特性。

例 7.3.2　计算图 7.3.2 所示网络中，输出变量 V_2 对频率 ω 的灵敏度以及此频率

下的群延时特性。设 $\omega = 1$。

解 原网络为

$$\begin{bmatrix} G_1 + G_2 + SC & -G_2 & 0 \\ -G_2 & -g_m + G_2 & 1 \\ 0 & 1 & -SL \end{bmatrix} \begin{bmatrix} V_1 \\ V_2 \\ I_L \end{bmatrix} = \begin{bmatrix} I_S \\ 0 \\ 0 \end{bmatrix}$$

解此方程,得到

图 7.3.2 例 7.3.2 电路

$$X = [V_1 \quad V_2 \quad I_L]^T = \left[\frac{6+2j}{20} \quad \frac{8+16j}{20} \quad \frac{16-8j}{20}\right]^T$$

伴随网络为

$$\begin{bmatrix} G_1 + G_2 + SC & -G_2 - g_m & 0 \\ -G_2 & G_2 & 1 \\ 0 & 1 & -SL \end{bmatrix} \begin{bmatrix} \hat{V}_1 \\ \hat{V}_2 \\ \hat{I}_L \end{bmatrix} = \begin{bmatrix} 0 \\ -1 \\ 0 \end{bmatrix}$$

解此伴随网络方程,得到

$$\hat{X} = [\hat{V}_1 \quad \hat{V}_2 \quad \hat{I}_L]^T = \left[\frac{-8-16j}{20} \quad \frac{-6-22j}{20} \quad \frac{-22+6j}{20}\right]^T$$

由于网络中只有两个电抗元件 C 和 L。由式(7.3.5),有

$$\frac{U_O}{\omega} = jCV_1\hat{V}_1 - jLI_L\hat{I}_L = \frac{384+288j}{400}$$

且

$$\tau = -\frac{\phi}{\omega} = -\operatorname{Im}\left[\frac{1}{U_O}\frac{U_O}{\omega}\right] = -\operatorname{Im}\left[\frac{20}{8+16j}\frac{384+288j}{400}\right]$$

$$= -\operatorname{Im}\frac{24-12j}{20} = \frac{12}{20}\ \text{rad}$$

7.3.3 SPICE 程序中的直流灵敏度分析

SPICE 程序可以计算电路的直流小信号灵敏度和交流小信号灵敏度(SPICE 3F 以上版本,才有交流小信号灵敏度)。直流小信号灵敏度是在完成直流工作点分析的基础上,用伴随网络法计算各指定输出变量对每个电路元件参数和器件模型参数的直流灵敏度,包括绝对灵敏度和相对灵敏度。绝对灵敏度是网络函数 T 对网络元器件参数 p 的简单偏导数:

$$S_p^T = \frac{T}{p} \tag{7.3.9}$$

S_p^T 称为绝对微分灵敏度,也叫非归一化灵敏度。SPICE 程序中称之为元件灵敏度(element sensitivity)。相对灵敏度(normalized sensitivity)在 SPICE 中表示为

$$S_p^T = \frac{T}{p}\frac{p}{100} \tag{7.3.10}$$

亦称之为归一化灵敏度。

SPICE 中直流小信号灵敏度分析的语句例子是:

.SENS V(1) V(2,4)

其中 V(1), V(2,4)是输出电压变量。该语句表示计算电路输出 V(1)和 V(2,4)对所有元器件参数的直流灵敏度。下面是一个简单的单管放大器电路的灵敏度分析实例,电路图如图 7.3.3 所示。

电路输入文件如下:

```
One  Transistor  Circuit
Q1   2   3   0   MOD1        ;图 7.3.3 中为 T1
RC   1   2   1K
RB   2   3   20K
VCC  1   0   6
.MODEL  MOD1  NPN  IS= 1E- 15  BF= 80  RB= 50  RE= 1
+   RC= 100  CJC= 2P  CJE= 2P  TF= 5NS  VAF= 100
.OP
.SENS  V(2)
.END
```

图 7.3.3 单管放大器

求出电路的直流工作点是:

NODE	VOLTAGE	NODE	VOLTAGE	NODE	VOLTAGE
(1)	6.0000	(2)	1.7911	(3)	.7581

V(2) 对各元件参数灵敏度的计算结果如下:

DC SENSITIVITIES OF OUTPUT V(2)

	ELEMENT NAME	ELEMENT VALUE	ELEMENT SENSITIVITY (VOLTS/UNIT)	NORMALIZED SENSITIVITY (VOLTS/PERCENT)
	RC	1.000E+03	-8.462E-04	-8.462E-03
	RB	2.000E+04	4.084E-05	8.167E-03
	VCC	6.000E+00	2.011E-01	1.206E-02
Q1 (图中为 T_1)	RB	5.000E+01	4.084E-05	2.042E-05
	RC	1.000E+02	3.426E-05	3.426E-05
	RE	1.000E+00	3.363E-03	3.363E-05
	BF	8.000E+01	-1.011E-02	-8.084E-03
	ISE	0.000E+00	0.000E+00	0.000E+00
	BR	1.000E+00	1.585E-11	1.585E-13
	ISC	0.000E+00	0.000E+00	0.000E+00
	IS	1.000E-15	-2.066E+13	-2.066E-04
	NE	1.500E+00	0.000E+00	0.000E+00
	NC	2.000E+00	0.000E+00	0.000E+00
	IKF	0.000E+00	0.000E+00	0.000E+00
	IKR	0.000E+00	0.000E+00	0.000E+00

VAF	1.000E+ 02	5.107E- 05	5.107E- 05
VAR	0.000E+ 00	0.000E+ 00	0.000E+ 00

由上述结果可以看出:外接电阻 R_B 的绝对灵敏度是 4.084E- 5,相对灵敏度是 8.167E- 3,这说明 R_B 每变化 1Ω,V(2)的电压升高 4.084E- 5 V,R_B 每变化 1%(200Ω),V(2)的电压升高 8.167E- 3 V。外接电阻 R_C 的灵敏度是负的,即 R_C 每变化 1Ω,V(2)电压下降 8.462E- 4 V。如果我们希望 V(2)的电位由其原静态工作点 1.7911V 降到 1.70V,我们可以通过调节 R_C 电阻来实现。由 1.7911V 降到 1.70V,其电压变化量是- 0.0911,用其除以 V(2)对 R_C 的绝对灵敏度得到 $\Delta R_C = \frac{-0.0911}{-8.462E-4} \approx 107.6\Omega$。按照这个结果,将 R_C 由 1k 增加到 1.1k,重新计算该电路的直流工作点,则得到 V(2)= 1.7126V。这说明可以用直流小信号灵敏度值来估算电路参数微小变化对输出特性的影响,并以此为据对电路元件参数做适当调整。当然,在此电路中,除了调整电阻 R_C 外,还可以调整电阻 R_B 或晶体管 T 的正向电流增益 β 等灵敏度较高的参数。

由于利用灵敏度信息指导电路设计十分有效,因此在电路优化设计中,利用电路元器件参数的灵敏度信息(也称之为梯度信息)去确定优化的搜索方向和步长。在这里我们以一个功放末级电路为例,介绍如何用灵敏度信息调整电路的直流电平。

图 7.3.4 所示是一个 OTL 电路,我们要求输出中点(A 点)电位为 6±0.05 V。

输入文件如下:

```
OTL Circuit
R1   1   2   2.7K
R2   3   4   2.4K
R3   3   9   18K
R4   3   8   1.8K
R5   6   7   27
R6   7   10  560
RL   10  0   8
C1   2   3   2.2U
C2   8   9   5600P
C3   9   10  220U
Q1   5   3   4   MOD1     ;图 7.3.4 中为 T1
Q2   4   5   9   MOD2     ;图 7.3.4 中为 T2
Q3   0   7   9   MOD3     ;图 7.3.4 中为 T3
D1   5   6   DMOD
.MODEL   MOD1  PNP  IS= 1E- 15  BF= 100
.MODEL   MOD2  NPN  IS= 1E- 13  BF= 80
.MODEL   MOD3  PNP  IS= 1E- 13  BF= 80
.MODEL   DMOD  D    IS= 1E- 14  RS= 50
VIN  1   0   AC  1     ;图 7.3.4 中为 EA
```

图 7.3.4 OTL 电路

```
VCC   4   0   12
.OP
.SENS   V(9)
.END
```

由 SPICE 程序计算出的直流工作点如下:

NODE	VOLTAGE	NODE	VOLTAGE	NODE	VOLTAGE	NODE	VOLTAGE
(1)	0.0000	(2)	0.0000	(3)	11.2370	(4)	12.0000
(5)	4.9324	(6)	3.9018	(7)	3.7253	(8)	11.2370
(9)	4.3324	(10)	.0525				

可以看出(9)节点(A 点)电位为 4.3324V, 不符合设计要求。我们可以根据下面计算出的灵敏度信息调整 A 点电位。

DC SENSITIVITIES OF OUTPUT V(9)

ELEMENT NAME	ELEMENT VALUE	ELEMENT SENSITIVITY (VOLTS/UNIT)	NORMALIZED SENSITIVITY (VOLTS/PERCENT)
R1	2.700E+03	0.000E+00	0.000E+00
R2	2.400E+03	1.760E-03	4.225E-02
R3	1.800E+04	-2.832E-04	-5.097E-02
R4	1.800E+03	0.000E+00	0.000E+00
R5	2.700E+01	7.192E-04	1.942E-04
R6	5.600E+02	1.717E-03	9.615E-03
RL	8.000E+00	1.717E-03	1.374E-04
VIN(图中为 E_A)	0.000E+00	0.000E+00	0.000E+00
VCC	1.200E+01	7.382E-01	8.859E-02

由上述列出的灵敏度值可知 R_2, R_3, R_6, R_L 的灵敏度较高, 但 R_L 是喇叭, 不能调; R_6 决定 T_3 管的偏置, 也不宜动; 所以我们可以通过增大 R_2 或减小 R_3 来提高 V(9)电位。例如我们将 R_2 值由 2.4k 增加到 3.9k, 则 V(9)电位升至 5.9862V, 符合设计要求。当然也可以通过减小 R_3 的阻值来提高 V(9)电位。R_2 改为 3.9k 后, 电路的直流工作点如下:

NODE	VOLTAGE	NODE	VOLTAGE	NODE	VOLTAGE	NODE	VOLTAGE
(1)	0.0000	(2)	0.0000	(3)	11.2280	(4)	12.0000
(5)	6.6794	(6)	5.5296	(7)	5.2928	(8)	11.2280
(9)	5.9862	(10)	.0745				

7.4 导数网络法

导数网络法也是用计算机求解电路元器件灵敏度的一种常用方法, 它是通过偏导数求微分灵敏度, 也称为增量网络法。这种方法通过求解导数网络方程, 可得到全部网络输

出量对一个网络参数变化的灵敏度值。

首先介绍什么是导数网络和导数网络方程。

设原网络方程为

$$TX = B \tag{7.4.1}$$

设 p 为求灵敏度时变化的某一参数。将网络方程(7.4.1)两边对 p 求导:

$$T\frac{X}{p} + \frac{T}{p}X = \frac{B}{p}$$

即

$$T\frac{X}{p} = \frac{B}{p} - \frac{T}{p}X \tag{7.4.2}$$

这就是导数网络方程。比较式(7.4.1)和式(7.4.2)可以看出,若以$\frac{X}{p}$与(7.4.1)式中的 X 对应,未知向量$\frac{X}{p}$就是网络的全部输出量对一个网络参数 p 的微分灵敏度;两个方程的系数矩阵相同,只是反映独立源贡献的右端向量不同。因此,我们可以设想构造一个导数网络 N_a,其特征如下:

(1) 导数网络 N_a 与原网络 N 有相同的拓扑结构,而且除独立源外的所有支路特性及数值也相同。

(2) 导数网络 N_a 以$\frac{I_b}{p}$,$\frac{U_b}{p}$和$\frac{V_n}{p}$为其支路电流、支路电压和节点电位。

(3) 导数网络方程的右端向量由两部分组成,一部分是$\frac{B}{p}$,反映网络中独立源的贡献;另一部分是$\frac{T}{p}X$,反映网络中产生变化的参数的特性。故导数网络 N_a 与原网络 N 在独立源的处理上有所不同。

下面我们通过对线性元件和非线性元件的分析,介绍导数网络和导数网络方程的形成,关键是导数网络方程右端向量的形成。

7.4.1 线性网络的导数网络法

1. 电导

电导的支路方程为

$$I_G = GU_G$$

将方程两端对参数 p 求导,有

$$\frac{I_G}{p} = G\frac{U_G}{p} + U_G\frac{G}{p}$$

当变化参数 p 不是电导 G,即 $p \neq G$ 时,有

$$\frac{I_G}{p} = G\frac{U_G}{p}$$

当变化参数 p 就是电导 G,即 $p = G$ 时,则

$$\frac{I_G}{G} = G\frac{U_G}{G} + U_G \tag{7.4.3}$$

这说明,当变化参数 p 不是电导 G 时,网络 N 中的 G 支路在导数网络 N_a 中仍为 G 支路;当 p = G 时,G 支路在导数网络 N_a 中除 G 支路本身外还再并联一个附加电流源,其值为 U_G。如图 7.4.1 所示。

图 7.4.1 (a) 原网络 N 中 G 支路; (b), (c) 导数网络 N_a 中 G 支路

上述推导说明,用导数网络法计算网络全部输出变量对某一参数 p 的灵敏度值,当参数 p 不是电导 G 时,导数网络中电导支路不变;当参数 p 就是 G 时,导数网络中电导支路上应并联一个附加电流源,该电流源对伴随网络方程的贡献是在方程右端向量中 G 支路对应位置上的送值。从下面的推导可以进一步看出,此结论适用于所有线性和非线性元件。

2. 电阻

电阻支路方程为

$$U_R = RI_R$$

对 p 求导:

$$\frac{U_R}{p} = R\frac{I_R}{p} + I_R\frac{R}{p}$$

当 $p \neq R$ 时,有

$$\frac{U_R}{p} = R\frac{I_R}{p}$$

当 p = R 时,有

$$\frac{U_R}{p} = R\frac{I_R}{p} + I_R$$

或改写为

$$\frac{I_R}{R} = \frac{1}{R}\frac{U_R}{R} - \frac{U_R}{R^2} \tag{7.4.4}$$

当网络中变化的参数是电阻 R 时,相当于在导数网络中的电阻 R 上并联一个电流源,其值为 $-U_R/R^2$。若变化参数不是电阻 R,则导数网络中 R 不变。如图 7.4.2 所示。

3. 独立源

(1) 独立电压源

独立电压源支路方程为

$$U_E = E$$

方程两边对 p 求导,得

图 7.4.2 (a) 原网络 N 中 R 支路; (b), (c) 导数网络 N_a 中 R 支路

$$\frac{U_E}{p} = \frac{E}{p}$$

当变化参数 $p \neq E$, 有

$$\frac{U_E}{p} = 0$$

当变化参数 $p = E$, 有

$$\frac{U_E}{p} = 1 \tag{7.4.5}$$

即在 N 中的独立电压源, 当 $p = E$ 时在 N_a 中为单位电压源; 当 $p \neq E$ 时, 为零值电压源, 即短路。如图 7.4.3 所示。

图 7.4.3 (a) 网络 N 中的独立电压源支路; (b), (c) 网络 N_a 中的独立电压源支路

(2) 独立电流源

独立电流源方程为

$$I_S = I_J$$

方程两边对 p 求导, 得

$$\frac{I_S}{p} = \frac{I_J}{p}$$

当 $p \neq I_J$ 时, 有

$$\frac{I_S}{p} = 0$$

当 $p = I_J$ 时, 有

$$\frac{I_S}{I_J} = 1 \tag{7.4.6}$$

即在 N 中的独立电流源,当 $p = I_J$ 时在 N_a 中为单位电流源;当 $p \neq I_J$ 时,为零值电流源,即开路。如图 7.4.4 所示。

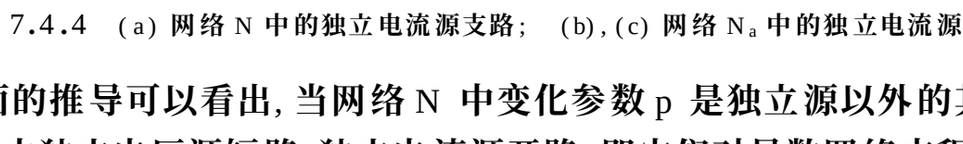


图 7.4.4 (a) 网络 N 中的独立电流源支路; (b),(c) 网络 N_a 中的独立电流源支路

通过上面的推导可以看出,当网络 N 中变化参数 p 是独立源以外的其它参数时,在导数网络 N_a 中独立电压源短路,独立电流源开路,即它们对导数网络方程右端向量没有贡献。当网络 N 中变化参数 p 就是独立源本身时,在导数网络 N_a 中,独立源是值为 1 的单位独立源。

4. 受控源

电流控制电流源 CCCS 的支路方程为

$$I_2 = \beta I_1$$

其中 I_1 和 I_2 分别是控制支路和被控支路的支路电流。将方程两端对参数 p 求导,得

$$\frac{I_2}{p} = \beta \frac{I_1}{p} + I_1 \frac{\beta}{p}$$

当 $p \neq \beta$ 时,$\frac{\beta}{p} = 0$,即在 N_a 中 CCCS 与原网络 N 中的一样,没有变化。

当 $p = \beta$ 时,$\frac{\beta}{p} = 1$,则

$$\frac{I_2}{\beta} = \beta \frac{I_1}{\beta} + I_1 \tag{7.4.7}$$

即在 N_a 中,CCCS 的被控制支路上要并联一个值为 I_1 的电流源。如图 7.4.5 所示。

图 7.4.5 (a) 网络 N 中的 CCCS 的支路; (b),(c) 网络 N_a 中的 CCCS 支路

其它三种受控源 VCCS,VCVS,CCVS,同样可以用与 CCCS 类似的方法,推导出其导数网络支路模型,在此我们就不一一赘述了。

稳态(包括直流和交流小信号)灵敏度分析的导数网络法的各线性元件支路模型归纳于表 7.4.1 中。

表 7.4.1　导数网络法线性元件模型

7.4.2 非线性网络的导数网络法

1. 非线性电导

非线性电导的支路方程是

$$I_g = f(U_g, p_g)$$

将方程两边对变化参数 p 求导，得

$$\frac{I_g}{p} = \frac{f}{U_g}\frac{U_g}{p} + \frac{f}{p_g}\frac{p_g}{p}$$

当 $p \neq p_g$ 时，有

$$\frac{I_g}{p} = \frac{f}{U_g}\frac{U_g}{p} \tag{7.4.8}$$

即在 N_a 中非线性电导等效为工作点附近的一个值为 $\frac{f}{U_g}$ 的电导。如图 7.4.6(b) 所示。

当 $p = p_g$ 时，有

$$\frac{I_g}{p} = \frac{f}{U_g}\frac{U_g}{p_g} + \frac{f}{p_g} \tag{7.4.9}$$

即在这种情况下，还要在电导 $\frac{f}{U_g}$ 上并联一个值为 $\frac{f}{p_g}$ 的电流源。如图 7.4.6(c) 所示。

图 7.4.6 (a) 网络 N 中非线性电导支路；(b), (c) 网络 N_a 中非线性电导支路

2. 非线性电阻

非线性电阻的支路方程是

$$U_R = \psi(I_R, p_R) \tag{7.4.10}$$

方程两边对 p 求导，得

$$\frac{U_R}{p} = \frac{\psi}{I_R}\frac{I_R}{p} + \frac{\psi}{p_R}\frac{p_R}{p}$$

当 $p \neq p_R$ 时，有

$$\frac{U_R}{p} = \frac{\psi}{I_R}\frac{I_R}{p}$$

即在 N_a 中等效为工作点附近的一个值为 $\frac{\psi}{I_R}$ 的电阻。如图 7.4.7(b) 所示。

当 $p = p_R$ 时，

$$\frac{U_R}{p} = \frac{\psi}{I_R}\frac{I_R}{p_R} + \frac{\psi}{p_R} \tag{7.4.11}$$

即在 N_a 中的电阻 $\frac{\psi}{I_R}$ 上串联一个值为 $\frac{\psi}{p_R}$ 的电压源。为了避免由于增加电压源而产生的附加节点，可将其转换为电流源，(7.4.11) 式可化为

$$\frac{I_R}{p_R} = 1/\frac{\psi}{I_R}\frac{U_R}{p_R} - \frac{\psi}{p_R}/\frac{\psi}{I_R} \tag{7.4.12}$$

图 7.4.7 (a) 网络 N 中的非线性电阻支路； (b), (c) 网络 N_a 中的非线性电阻支路

相当于值为 $1/\frac{\psi}{I_R}$ 的电导与值为 $-\frac{\psi}{p_R}/\frac{\psi}{I_R}$ 的电流源并联。如图 7.4.7(c) 所示。

7.4.3 导数网络方程的建立与求解

由前面所推导的线性元件和非线性元件的导数网络模型可以看出，原网络 N 和导数网络 N_a 的区别只是独立源的位置和数值不同，两网络中的阻抗或导纳(包括受控源的控制参数)的位置和数值均不变。因此，导数网络方程系数矩阵与原网络方程的系数矩阵相同，而反映独立源贡献的右端向量不同。若原网络 N 的电路方程为

$$TX = B$$

则导数网络方程的形式为

$$T\frac{X}{p} = B_a \tag{7.4.13}$$

由于式(7.4.13)中未知向量 $\frac{X}{p}$ 表示网络的全部输出量 X 对某一网络参数 p 的灵敏度，故右端向量 B_a 中最多只含有与参数 p 有关的两个非零元项，即由于参数 p 变化在网络 N_a 中增加的电流源的贡献。各元件参数变化在 N_a 中增加的电流源的数值可以从表 7.4.1 中查到。

因为导数网络方程的系数矩阵与原网络系数矩阵相同，因而在原网络方程求解过程中系数矩阵的 LU 分解结果，可以在导数网络方程的求解中直接使用，故求解导数网络方程所需乘除运算量仅是向前、向后替代所需的乘除次数，比求解原网络方程的运算量小得多。我们将原网络方程系数矩阵 T 进行 LU 分解：

$$T = LU$$

由此建立导数网络方程

$$LU\frac{X}{p} = B_a \tag{7.4.14}$$

令
$$U\frac{X}{p}=\frac{y}{p} \tag{7.4.15}$$
则有
$$L\frac{y}{p}=B_a$$
导数网络法求稳态灵敏度的步骤归纳如下:

(1) 对原网络 N 进行稳态分析, 求解原网络方程 TX = B, 并保留系数矩阵的 LU 分解结果。

(2) 建立导数网络方程
$$T\frac{X}{p}=B_a$$
其系数矩阵 T 与原网络系数矩阵相同, 不必重新组成。

对一个变化的参数 p, 按表 7.4.1 所示模型, 向导数网络方程的右端向量 B_a 送值, 构成导数网络方程。

(3) 利用原网络方程系数矩阵的 LU 分解结果, 求解导数网络方程, 得到全部网络输出量对一个变化参数 p 的灵敏度值 $\frac{X}{p}$。

(4) 每一个变化参数形成一个新的右端向量, 即一个导数网络方程。若要求输出量对多个元件参数的灵敏度, 必须求解多个导数网络方程。

由于人们感兴趣的往往是少数输出量对多个元件参数的灵敏度, 而不是全部输出量对一个或几个元件参数的灵敏度, 故在这一点上导数网络法不如伴随网络法。但是在瞬态灵敏度分析中, 一般采用导数网络法, 因我们在本章中不涉及瞬态灵敏度的内容, 故不在此叙述导数网络法用于瞬态灵敏度分析的原理及优点。

图 7.4.8　例 7.4.1 电路

例 7.4.1　用导数网络法求图 7.4.8 电路中直流灵敏度 $\frac{X}{E}$, $\frac{X}{R_1}$。

解　原网络方程 TX = B 为
$$\begin{bmatrix} \frac{1}{3} & -\frac{1}{3} & 0 & 1 \\ -\frac{1}{3} & \frac{1}{3}+\frac{1}{2}+1 & -1 & 0 \\ 0 & -1 & 1+1 & 0 \\ 1 & 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} V_1 \\ V_2 \\ V_3 \\ I_E \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ 0 \\ 10 \end{bmatrix}$$
求解此方程, 得到
$$X = [V_1 \quad V_2 \quad V_3 \quad I_E]^T = \begin{bmatrix} 10 & \frac{5}{2} & \frac{5}{4} & -\frac{5}{2} \end{bmatrix}^T$$

(1) 求 $\frac{X}{E}$

导数网络方程的系数矩阵即为原网络方程系数矩阵 T, 变化参数为电压源 E, 由表

7.4.1 查得 E 变化而增加的电压源值为单位 1, 故导数网络方程为

$$\begin{bmatrix} \frac{1}{3} & -\frac{1}{3} & 0 & 1 \\ -\frac{1}{3} & \frac{11}{6} & -1 & 0 \\ 0 & -1 & 2 & 0 \\ 1 & 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} V_1/E \\ V_2/E \\ V_3/E \\ I_E/E \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ 0 \\ 1 \end{bmatrix}$$

解此导数网络方程, 得

$$\frac{X}{E} = \left[\frac{V_1}{F} \quad \frac{V_2}{E} \quad \frac{V_3}{E} \quad \frac{I_E}{E}\right]^T = \left[1 \quad \frac{1}{4} \quad \frac{1}{8} \quad -\frac{1}{4}\right]^T$$

(2) 求 $\frac{X}{R_1}$

由表 7.4.1 查得 R_1 变化而增加的电流源值为 $-\frac{U_{R_1}}{R_1^2} = -\frac{V_1 - V_2}{R_1^2} = -\frac{7.5}{9}$, 故导数网络方程为

$$\begin{bmatrix} \frac{1}{3} & -\frac{1}{3} & 0 & 1 \\ -\frac{1}{3} & \frac{11}{6} & -1 & 0 \\ 0 & -1 & 2 & 0 \\ 1 & 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} V_1/R_1 \\ V_2/R_1 \\ V_3/R_1 \\ I_E/R_1 \end{bmatrix} = \begin{bmatrix} \frac{7.5}{9} \\ -\frac{7.5}{9} \\ 0 \\ 0 \end{bmatrix}$$

其解为

$$\frac{X}{R_1} = \left[\frac{V_1}{R_1} \quad \frac{V_2}{R_1} \quad \frac{V_3}{R_1} \quad \frac{I_E}{R_1}\right]^T = \left[0 \quad -\frac{5}{8} \quad -\frac{5}{16} \quad \frac{5}{8}\right]^T$$

例 7.4.2　用导数网络法求图 7.4.9 电路的输出向量对电阻 R 及二极管饱和电流 I_S 的灵敏度。已知二极管电流与电压关系式为 $I_D = I_S(e^{U_D/V_T} - 1)$, 其中 $I_S = 2.5 \times 10^{-13}$A, $V_T = 0.026$V。

解　原网络方程组为

$$\begin{bmatrix} \frac{1}{R} & -\frac{1}{R} & 1 \\ -\frac{1}{R} & \frac{1}{R} + G_{DM}^k & 0 \\ 1 & 0 & 0 \end{bmatrix} \begin{bmatrix} V_1 \\ V_2 \\ I_E \end{bmatrix} = \begin{bmatrix} 0 \\ -I_{DM}^k \\ E \end{bmatrix}$$

图 7.4.9　例 7.4.2 电路

这里 G_{DM}^k, I_{DM}^k 是采用 N-R 方法线性化的二极管直流伴随模型参数, 有

$$G_{DM}^k = \frac{I_D}{U_D} = \frac{I_S}{V_T} e^{U_D^k/V_T}$$

$$I_{DM}^k = I_S\left(e^{U_D^k/V_T} - 1\right) - G_{DM}^k U_D^k$$

经过 N-R 迭代, 此方程的解为

$$X = [V_1 \quad V_2 \quad I_E]^T = [10 \quad 0.604 \quad -3.48 \times 10^{-3}]^T$$

(1) 求$\frac{X}{R}$

R 是线性参数,与非线性器件 D 无关,由表 7.4.1 查得 R 变化增加的电流源值为 $-\frac{U_R}{R^2}=-\frac{9.40}{(2.7\times 10^3)^2}$,故导数网络方程为

$$\begin{bmatrix} \frac{1}{R} & -\frac{1}{R} & 1 \\ -\frac{1}{R} & \frac{1}{R}+G_{DM} & 0 \\ 1 & 0 & 0 \end{bmatrix} \begin{bmatrix} V_1/R \\ V_2/R \\ I_E/R \end{bmatrix} = \begin{bmatrix} \frac{U_R}{R^2} \\ -\frac{U_R}{R^2} \\ 0 \end{bmatrix}$$

其中 $G_{DM}=\frac{I_D}{U_D}=\frac{I_S}{V_T}e^{U_D/V_T}=\frac{2.5\times 10^{-13}}{0.026}e^{0.604/0.026}=0.118$,是工作点上的一个线性电导。故有

$$\begin{bmatrix} \frac{1}{2.7\times 10^3} & -\frac{1}{2.7\times 10^3} & 1 \\ -\frac{1}{2.7\times 10^3} & \frac{1}{2.7\times 10^3}+0.118 & 0 \\ 1 & 0 & 0 \end{bmatrix} \begin{bmatrix} V_1/R \\ V_2/R \\ I_E/R \end{bmatrix} = \begin{bmatrix} \frac{9.40}{(2700)^2} \\ -\frac{9.40}{(2700)^2} \\ 0 \end{bmatrix}$$

解得

$$\frac{X}{R}=\begin{bmatrix} \frac{V_1}{R} & \frac{V_2}{R} & \frac{I_E}{R} \end{bmatrix}^T=[0 \quad -9.55\times 10^{-6} \quad 1.285\times 10^{-6}]^T$$

(2) 求$\frac{X}{I_S}$

这里 I_S 是非线性器件二极管的参数,由式(7.4.10)知,在线性电导 G_{DM} 上并联了电流值为$\frac{I_D}{I_S}=e^{U_D/V_T}-1$ 的电流源。故导数网络方程为

$$\begin{bmatrix} \frac{1}{2.7\times 10^3} & -\frac{1}{2.7\times 10^3} & 1 \\ -\frac{1}{2.7\times 10^3} & \frac{1}{2.7\times 10^3}+0.118 & 0 \\ 1 & 0 & 0 \end{bmatrix} \begin{bmatrix} \frac{V_1}{I_S} \\ \frac{V_2}{I_S} \\ \frac{I_E}{I_S} \end{bmatrix} = \begin{bmatrix} 0 \\ -1.23\times 10^{10} \\ 0 \end{bmatrix}$$

解得

$$\frac{X}{I_S}=\begin{bmatrix} \frac{V_1}{I_S} & \frac{V_2}{I_S} & \frac{I_E}{I_S} \end{bmatrix}^T=[0 \quad -1.032\times 10^{11} \quad -3.821\times 10^{7}]^T$$

习　题

7.1　用伴随网络法求题图 7.1 所示网络中 U_o 对全部电导和电流源的灵敏度。

7.2　用伴随网络法求题图 7.2 所示网络中输出电压 U_o 对跨导 g_m($g_m=1S$)的灵敏度。

题图 7.1

题图 7.2

7.3 用伴随网络法求题图 7.3 所示网络中 V_2 对所有元件的灵敏度，令 $\omega = 1\text{rad/s}$。

题图 7.3

7.4 用伴随网络法求题图 7.4 中输出(U_o 或 I_o)对全部元件的灵敏度，令 $\omega = 1\text{rad/s}$。

题图 7.4

7.5 用导数网络法求题图 7.2 中网络的输出向量对所有网络元件参数的灵敏度值。

7.6 用导数网络法求题图 7.4 中各网络的输出向量对所有网络元件参数的灵敏度值。

7.7　已知题图 7.7 所示电路的静态工作点为

$$[V_1 \quad V_2 \quad I_E]^T = [2.00 \quad 1.48 \quad -1.48E-3]^T$$

用伴随网络法求 V_2 对 E, R_L 以及二极管参数 I_S, n(发射系数) 和 V_T 的灵敏度。令二极管参数为: $I_S = 10^{-10}$A, $R_S = 0$, $n = 1.2$, $V_T = \dfrac{nkT}{q}$, $E = 2$V, $R_L = 1$kΩ。

题图　7.7

7.8　推导伴随网络法的受控源 VCCS 和 CCCS 的灵敏度公式。

第8章　容差分析

8.1　引　　言

任何电子产品在制造和应用过程中都不可避免地受到随机扰动因素的影响，因此实际电路中的元器件参数和其标称值之间总是存在着随机的误差，也就是说电路中的元器件参数的数值是其标称值容差范围内的一个随机数值。例如，一个标称值为 47kΩ，容差范围是 10% 的电阻器，其阻值可以是 42.3kΩ到 51.7kΩ范围内的任何数。在电路设计过程中，一般采用元器件参数的标称值进行电路仿真，计算电路的性能。由此计算出的电路与实际生产制造出的电路必然存在差别，而且随着环境温度的变化和使用时间的增加，这种差别还可能进一步增大。在严重的情况下，电路性能的偏差可能达到影响电路正常使用的程度。因此，电子产品的设计必须考虑元器件参数容差的影响。

通常，容差问题包括容差设计和容差分析两部分内容。容差设计的任务是设计电路的标称值及分配电路中元器件参数的容差，使电路性能的偏差最小；或者在保证电路性能满足指标要求的条件下允许元器件参数的容差范围最大，所以容差设计问题也称之为电路参数的容差分配设计。而且，元器件参数的容差不仅关系到电路的性能指标，也与电路的经济成本直接相关。通常，元器件的精度越高其误差范围越小，价格也就越高。我们希望在电路性能满足要求，价格尽可能低的情况下，实现电路参数的最佳设计。容差分析是在给定电路参数容差范围的条件下，计算器件参数变化对电路性能的影响。这一章我们主要讨论容差分析问题。

对电路进行容差分析有两类方法。一类是以灵敏度分析为基础的方法。灵敏度分析只是解决单个元器件参数变化对电路性能的影响，而容差分析是利用灵敏度信息解决多个元器件参数偏离标称值对电路性能的总影响。例如我们在这章里介绍的最坏情况分析。第二类方法是统计方法。在很多情况下，我们并不可能确切知道每个器件参数实际偏离标称值的量，只是知道它们的可能偏离范围和各个参数的随机分布规律。我们可以利用概率统计的方法，通过已知的器件参数的随机分布规律去计算电路特性的分布规律。蒙特卡罗分析就是一种统计抽样方法。这种方法在器件参数容差范围内对参数进行随机抽样，对大量的抽样值做电路仿真，计算出电路性能的统计特性和偏差范围。

最后说明一点：估计电路的合格率及合格率的梯度也属于容差分析的范畴，这个问题我们放到下一章电路优化问题中去讨论。

在详细讨论容差分析问题之前，首先对器件参数的统计分布规律的特性作一简单的介绍。

8.2 器件参数的统计分布规律

8.2.1 随机变量及其描述方法

假定电路中某一个电阻的标称值为 100Ω, 容差为 5%。我们对于总数 N = 1000 的这种电阻逐个地进行测量, 每个电阻的阻值可能是 95 至 105Ω之间的任何值。将 95 至 105Ω的区间分成间隔为 1Ω的 10 个子区间, 记录阻值落入每一个子区间的电阻个数, 并将测量结果列于表 8.2.1 中。表 8.2.1 数据所反映的电阻的统计分布也可以画成图 8.2.1(a)所示的直方图, 图中横坐标是划分为 10 个子区间的电阻值, 纵坐标为测得的阻值落入第 i 个子区间的电阻个数 N_i。图 8.2.1(b)表示这 1000 个电阻的概率密度分布, 它与图 8.2.1(a)相似, 只是将纵坐标的量纲换成相对频率 $n_i = N_i / N$。对这种通过抽样测量估计其统计分布规律的离散随机变量, 用直方图表示其统计分布规律是最简单也是最直观的一种方式。

表 8.2.1 1000 个标称值为 100Ω, 容差为 5% 的电阻的测量结果

区间 i	1	2	3	4	5	6	7	8	9	10
阻值	95～96	96～97	97～98	98～99	99～100	100～101	101～102	102～103	103～104	104～105
个数 N_i	22	50	101	161	192	180	141	111	32	10

图 8.2.1 (a) 标称值 100Ω电阻的直方图; (b) 标称值 100Ω电阻的概率密度分布图

我们以这个电阻为例, 说明如何用一个离散随机变量描述电路元器件参数的统计分布规律。假定 X 是个离散随机变量, 事件 $X = x_i$ 表示测量一个电阻的阻值落在第 i 个子区间。显然, 事件$\{X = x_i\}$的概率可近似为

$$P\{X = x_i\} = p_i \approx N_i / N \quad i = 1, 2, \cdots \tag{8.2.1}$$

如果抽样点数 N 足够大, (8.2.1)式的近似就足够精确。我们称(8.2.1)式为离散随机变量 X 的概率分布或分布律。

由以上的概率定义, p_i 应满足如下两个条件:

$$p_i \geq 0 \qquad i = 1, 2, \cdots$$
$$\sum_{i=1}^{\infty} p_i = 1 \tag{8.2.2}$$

离散随机变量 X 的概率描述了电阻值的分布规律，其对应关系如表 8.2.2 所示。抽样点数越多。区间划分越细，描述越准确。

表 8.2.2　离散随机变量与阻值的对应关系

阻值	95～96	96～97	97～98	98～99	99～100	100～101	101～102	102～103	103～104	104～105
x_i	95	96	97	98	99	100	101	102	103	104
p_i	0.022	0.050	0.101	0.161	0.192	0.18	0.141	0.111	0.032	0.010

实际上，电路中的器件参数可以取其容差范围内的任意数值，而不只是区间内的离散值，这时要用器件参数的概率密度函数或概率分布函数描述其统计分布特性。假定 X 是一个连续型随机变量，$F(x) \geq 0$，x 是任意实数，函数

$$F(x) = P\{X \leq x\} \tag{8.2.3}$$

称为 X 的分布函数。

如果将 X 看成数轴上随机点的坐标，那么分布函数 $F(x)$ 在 x 处的函数值就表示 X 落在区间 $(-\infty, x]$ 上的概率。

分布函数具有如下性质：

(1) $0 \leq F(x) \leq 1$，$F(-\infty) = 0$，$F(\infty) = 1$；

(2) $F(x)$ 单调不减，即若 $x_1 < x_2$，则有 $F(x_1) \leq F(x_2)$。

如果存在函数 $p(x) \geq 0$，使对于任意实数 x 有

$$F(x) = \int_{-\infty}^{x} p(t)\,dt \tag{8.2.4}$$

则称 $p(x)$ 为随机变量 X 的概率密度函数，简称密度函数。概率密度函数具有如下性质：

(1) $\int_{-\infty}^{\infty} p(x)dx = 1$

(2) $P(x_1 < X \leq x_2) = \int_{x_1}^{x_2} p(x)dx = F(x_2) - F(x_1)$

(3) $p(x) = \dot{F}(x)$

概率分布函数和概率密度函数都可以完备描述随机变量，这就是说，知道了电路中器件参数的分布函数或密度函数，就可以完备描述器件参数的统计分布特征，从而对电路进行容差分析。

8.2.2　随机变量的数字特征

虽然分布函数能够完整地描述随机变量的统计特性，但在实际问题中求随机变量的分布函数决非易事。好在许多情况下只需知道随机变量的某些特征就足够了。随机变量常用的数字特征是：数学期望、方差和矩。

对于离散分布的随机变量 X，其概率分布为

$$P\{X = x_i\} = p_i \quad i = 1, 2, \cdots \tag{8.2.5}$$

则其数学期望 μ_k 和方差 σ^2 分别定义为

$$\mu_k = E(X) = \sum_{i=1}^{\infty} x_i p_i \tag{8.2.6}$$

$$\sigma^2 = D(X) = \sum_{i=1}^{\infty} [x_i - E(X)]^2 p_i \tag{8.2.7}$$

其中，μ_k 称为随机变量 X 的数学期望，简称期望或均值；σ 称为随机变量的标准差或均方差。

对于连续随机变量 X，若其概率密度为 $p(x)$，则数学期望和方差的定义分别为

$$\mu_k = E(X) = \int_{-\infty}^{\infty} xp(x)dx \tag{8.2.8}$$

$$\sigma^2 = D(X) = \int_{-\infty}^{\infty} [x - E(X)]^2 p(x)dx \tag{8.2.9}$$

数学期望有如下几个重要性质：

(1) 设 c 为一常数，则 $E(c) = c$。

(2) 设 X_i 为随机变量，c_i 为常数，$i = 1, 2, \cdots, n$，则

$$E(c_1X_1 + \cdots + c_nX_n) = c_1E(X_1) + \cdots + c_nE(X_n)$$

(3) 若随机变量 X_i，$i = 1, 2, \cdots, n$ 相互独立，则

$$E(X_1X_2\cdots X_n) = E(X_1)E(X_2)\cdots E(X_n)$$

随机变量的方差有如下几个重要性质：

(1) 若 c 为常数，则 $D(c) = 0$。

(2) $D(X) = E(X^2) - [E(X)]^2$。

(3) 设 X 和 Y 都是随机变量，则

$$D(X + Y) = D(X) + D(Y) + 2E[X - E(X)]E[Y - E(Y)]$$

上式中最后一项称作随机变量 X 和 Y 的协方差，即

$$\mathrm{cov}(X, Y) = E[X - E(X)]E[Y - E(Y)] \tag{8.2.10}$$

若随机变量 X 和 Y 的方差全不为零，则可以根据协方差定义它们的相关系数 ρ

$$\rho = \frac{\mathrm{cov}(X, Y)}{\sqrt{D(X)D(Y)}} \tag{8.2.11}$$

若两个随机变量相互独立，则它们的协方差为零；反之，如果两个随机变量的协方差等于零，它们不一定相互独立。

随机变量还有一个重要的数字特征是矩。设 X 和 Y 是随机变量，则

(1) X 的 k 阶矩为 $E(X^k)$，$k = 1, 2, \cdots$

(2) X 的 k 阶中心矩为 $E\{[X - E(X)]^k\}$，$k = 1, 2\cdots$

(3) X 和 Y 的 $k + l$ 阶混合矩为 $E(X^kY^l)$，$k, l = 1, 2, \cdots$

(4) X 和 Y 的 $k + l$ 阶中心混合矩为 $E\{[X - E(X)]^k[Y - E(Y)]^l\}$，$k, l = 1, 2, \cdots$

显然，X 的数学期望 $E(X)$ 是 X 的一阶矩，方差 $D(X)$ 是二阶中心矩；协方差 $\mathrm{cov}(X, Y)$ 是 X 和 Y 的二阶中心混合矩。

例 8.2.1　对概率分布如表 8.2.2 所示的随机变量 X，计算其数学期望和标准差。

解　由(8.2.6)式,有

$$\mu_X = \sum_{i=1}^{10} x_i p_i = 99.372$$

由(8.2.7)式,有

$$\sigma_X = \sqrt{D(X)} = \sqrt{\sum_{i=1}^{10} [x_i - E(X)]^2 p_i} = \sqrt{3.733} = 1.932$$

由此可见,用离散随机变量描述电路器件参数的统计分布规律时,选择适当的抽样点数、区间划分的边界以及随机变量的取值是至关重要的。

8.2.3　几种常用器件参数的统计分布

1. 正态分布

最常见的器件参数的统计分布是正态分布,也称高斯分布。各种电阻器、电容器以及晶体管的参数通常都服从正态分布。另外,根据大数定理,一定数量的器件组成的电路,不论器件参数的分布规律如何,电路的响应也近似正态分布。因此,我们应对正态分布的随机变量的特性比较熟悉。

设随机变量 X 服从正态分布,则其概率密度函数为

$$p(x) = \frac{1}{\sigma\sqrt{2\pi}} e^{-(x-\mu)^2/2\sigma^2}, \quad -\infty < x < \infty \tag{8.2.12}$$

其中,$\sigma > 0$,μ和 σ全为常数。该分布也可以记作 $N(\mu, \sigma^2)$,经过简单变量代换 $y = (x-\mu)/\sigma$可以将 $N(\mu, \sigma^2)$变换成标准正态分布 $N(0,1)$,即有

$$p(x) \quad y=(x-\mu)/\sigma \quad p(y) = \frac{1}{\sqrt{2\pi}} e^{-y^2/2} \tag{8.2.13}$$

由式(8.2.12)和式(8.2.13)可知,不同的 μ值,相应于概率密度函数曲线的平移;当 σ增大时(比如由 σ 增为 σ),曲线峰值下降而分布变宽,这种关系如图 8.2.2 所示。

图 8.2.2　正态分布的概率密度函数

正态分布的随机变量有如下性质:

(1) 数学期望: $E(X) = \int_{-\infty}^{\infty} x p(x) dx = \mu$。

(2) 方差: $D(X)=\int_{-\infty}^{\infty}(x-\mu)^2 p(x)\,dx=\sigma^2$。当 σ小时,随机变量的离散度小,分布集中在均值的附近; 当 σ大时,随机变量的离散度大,分布曲线展宽。

(3) 概率密度函数关于 μ值对称,即 $p(x-\mu)=p(-(x-\mu))$。

(4) 概率密度函数的峰值点在 μ处,即

$$\{\max_x p(x)\}=p(\mu)=1/(\sqrt{2\pi}\sigma)\approx 0.4/\sigma$$

应该指出,对于正态分布的随机变量,随机变量的取值偏离均值 3σ之外的概率是相当小的。假定随机变量 X 的取值落在其均值的 kσ邻域之内的概率为

$$P\{k\sigma\}=P\{\mu-k\sigma\leq X<\mu+k\sigma\}=\int_{\mu-k\sigma}^{\mu+k\sigma}\frac{1}{\sqrt{2\pi}\sigma}e^{-(x-\mu)^2/2\sigma^2}dx=2\int_0^k\frac{1}{\sqrt{2\pi}}e^{-x^2/2}dx$$

表 8.2.3 列出了不同的 k 值对应的 P{kσ}值。可以看出,正态分布随机变量的取值偏离其均值 3σ之外的概率只有 0.3%,所以通常将 3σ作为正态分布随机变量的容差界。

表 8.2.3 正态分布随机变量不同 k 值下的 P{kσ}值

k 值	1	1.64	1.96	2.0	2.33	2.58	2.81	3.0	3.09	3.29
P{kσ}值	68%	90%	95%	95.4%	98%	99%	99.5%	99.7%	99.8%	99.99%

2. 均匀分布

均匀分布是一种非常有用的统计分布,许多其它分布的随机变量都可以由均匀分布通过变换而得到。电路中有些器件参数的分布在特定情况下也可以认为是均匀分布的。设 X 是服从于均匀分布的随机变量,则其概率密度函数在其容差界之间是均匀的,即

$$p(x)=\begin{cases}\dfrac{1}{x_U-x_L} & x_L\leq x\leq x_U\\ 0 & x<x_L,\quad x>x_U\end{cases}\qquad(8.2.14)$$

其概率密度函数如图 8.2.3 所示。

图 8.2.3 均匀分布的概率密度函数曲线

容易算出均匀分布随机变量的数学期望和方差分别为

$$\mu=E(X)=\frac{x_U+x_L}{2}\qquad(8.2.15)$$

$$\sigma^2=D(X)=\frac{(x_U-x_L)^2}{12}=\frac{(x_U-\mu)^2}{3}=\frac{(\mu-x_L)^2}{3}\qquad(8.2.16)$$

3. 经参数筛选的正态分布

由于某些原因,器件在出厂以前可能会经过筛选,比如将参数最接近标称值的器件选出来作为高档产品,或将参数容差偏离标称值远的器件舍去不用。图 8.2.4(a)和(b)分别表示了上述这两种分布。这些分布整体上还是正态分布,只是经过筛选舍去了某些部分。

图 8.2.4　正态分布器件经参数筛选余下部分的概率密度曲线

8.3　多个器件参数变化对电路性能的影响

容差分析是估算所有器件参数都偏离其标称值的情况下电路性能的变化。解决这个问题的最简单的方法是先对电路性能作线性近似，然后利用灵敏度分析的结果估算电路性能的偏移量。

设电路中器件参数矢量为 $x=(x_1, x_2, \cdots, x_n)^T$，电路性能矢量为 $f=(f_1, f_2, \cdots, f_m)^T$，假定器件参数为标称值时的参数矢量为 $x^0=(x_1^0, x_2^0, \cdots, x_n^0)^T$，在器件参数标称值情况求得的电路性能矢量为 $f^0=(f_1^0, f_2^0, \cdots, f_m^0)^T$。将性能矢量在 x^0 点展开成台劳级数，略去二阶以上高阶项，可以得到器件参数变化 $\Delta x=x-x^0$ 时，电路性能的近似变化量 Δf，即有

$$\Delta f = f - f^0 = (\grave{e} f \mathbin{\copyright}_{x=x^0}) \Delta x \tag{8.3.1}$$

其中

$$\grave{e} f = \begin{bmatrix} \dfrac{f_1}{x_1} & \dfrac{f_1}{x_2} & \cdots & \dfrac{f_1}{x_n} \\ \dfrac{f_2}{x_1} & \dfrac{f_2}{x_2} & \cdots & \dfrac{f_2}{x_n} \\ \vdots & \vdots & \vdots & \vdots \\ \dfrac{f_m}{x_1} & \dfrac{f_m}{x_2} & \cdots & \dfrac{f_m}{x_n} \end{bmatrix}_{m\times n} = [\grave{e} f_1 \quad \grave{e} f_2 \quad \cdots \quad \grave{e} f_m]^T \tag{8.3.2}$$

由(8.3.2)式可以看出，$\grave{e} f$ 是一个一阶偏导数矩阵，它表示电路的全部输出变量对所有器件参数的灵敏度，也称之为梯度。因此，我们在对电路进行灵敏度分析的基础之上，就可以用(8.3.1)式估计电路中多个器件参数发生变化时电路输出性能的变化量。应该注意的是，只有在所有器件参数的变化量 Δx_i 都很小的条件下，(8.3.1)式才是正确的；当 Δx_i 较大时，(8.3.1)式不能忽略高阶项的影响。

当我们只对电路中某一个或某几个输出变量有兴趣时，(8.3.2)式所表示的梯度矩阵的维数可大大降低。

例 8.3.1　差分放大器如图 8.3.1 所示,两个晶体管的参数为 β = 60, I_S = 1E-13A,假设 5 个电阻的数值发生 5% 的变化,试估算节点电位 V(4)、V(5)和 V(4)- V(5)的变化量。

图 8.3.1　差分放大器

解　(1) 利用电路模拟程序 PSpice 对该差分对电路进行直流工作点分析,输出结果如下:

NODE	VOLTAGE	NODE	VOLTAGE	NODE	VOLTAGE	NODE	VOLTAGE
(1)	-.0259	(2)	-.0259	(3)	-.6328	(4)	7.8072
(5)	7.8072	(6)	12.0000	(7)	-12.0000		

由此可以看出,在器件参数为标称值情况下,节点④、⑤的电位 V(4)、V(5)的值为 7.8072V。

(2) 分别以 V(4)、V(5)和 V(4)- V(5)为输出变量,用 PSpice 程序计算 5 个电阻的灵敏度值,其结果列于表 8.3.1。

表 8.3.1　对图 8.3.1 所示的差分放大器进行容差分析的结果

f	f/R_{B1}	f/R_{B2}	f/R_E	f/R_{C1}	f/R_{C2}	Δf
V(4)	1.053E-3	-1.044E-3	1.159E-3	-1.553E-3	-6.468E-11	-5.85E-4
V(5)	-1.044E-3	1.053E-3	1.159E-3	-6.468E-11	-1.553E-3	-5.85E-4
V(4)- V(5)	2.097E-3	-2.097E-3	6.017E-20	-1.55E-3	1.553E-3	1.83E-17

(3) 假定 5 个电阻的阻值都增大 5%,利用前面计算出的灵敏度信息,按照(8.3.1)式估算出 V(4)、V(5)和 V(4)- V(5)的变化量,并列于表 8.3.1 的 Δf 项中。容易看出,如果电阻 R_{C1}和 R_{C2}同时增加(减少)同样的数值,或电阻 R_{B1}和 R_{B2}同时增加(减少)同样的数值,对 V(4)- V(5)的数值不会产生影响。所以,如果差分电路中差分对管的参数对称性较好,则用 V(4)- V(5)作为输出,其值较为稳定。

8.4 最坏情况分析

顾名思义，最坏情况是指，电路中器件参数在其容差范围内取某种组合时，所引起电路性能的最大偏差。我们在已知电路器件参数的容差而不知道其统计分布规律的条件下，可以通过最坏情况分析估算出电路性能的这种最坏偏差。

假定电路中器件参数的标称值矢量为 $x^0=(x_1, x_2, \cdots, x_n)^T$ 器件参数的容差矢量为 $t=(t_1, t_2, \cdots, t_n)^T$, $t_i>0$, 那么，满足条件

$$R_T=\{x \mid x_i^0-t_i \leq x \leq x_i^0+t_i,\ i=1, 2, \cdots, n\} \tag{8.4.1}$$

的器件参数矢量就构成了器件参数空间的容差域。图 8.4.1(a)、(b)分别为一维参数空间和二维参数空间中容差域的示意图。

图 8.4.1 参数空间及容差域

(a) 一维参数空间； (b) 二维参数空间

所谓最坏情况分析，就是在器件参数满足 $x \in R_T$ 的条件下，求电路性能偏离其标称值的最大量，即可以用下述两个数学规划来描述：

$$f^{max}=\max_{x \in R_T} f(x) \tag{8.4.2}$$

和

$$f^{min}=\min_{x \in R_T} f(x)$$

最坏情况问题(8.4.2)的求解，特别是全局最坏情况的求解，是个非常困难的数学问题，至今还没有对所有电路形式都适用的算法，本节只介绍一些最简单的方法。

我们以一维参数空间为例，假定电路中器件参数为 x_1 和 x_2，电路的特性函数 f_j 随 x_1 和 x_2 变化的情况分别如图 8.4.2(a)、(b)所示。我们将 $f_j(x_i)$, $i=1, 2$ 在 x_i^0 点线性化，即过 x_i^0 点做 $f_j(x_i)$的切线 AB，用直线 AB 近似函数 $f_j(x_i)$。可以看出，直线 AB 的最坏情况发生在容差域的边界处(函数 f_j 的最坏情况也发生在容差域的边界处)。因此，可以利用器件参数容差的边界值和灵敏度(切线的斜率)信息求得电路特性函数的最坏情况。比如，图 8.4.2(a)中在 x_1 的标称值 x_1^0 处 f_j 对 x_1 的灵敏度(或梯度)值为正，而在图 8.4.2(b)中 x_2 的标称值 x_2^0 处 f_j 对 x_2 的灵敏度为负，则求 f_j 的最大值 f_j^{max}，就应取 x_1 的最大值 $x_1^h=x_1^0+t_1$，取 x_2 的最小值 $x_2^l=x_2^0-t_2$，即

$$f_j^{max} = f_j(x_1^h, x_2^l)$$

图 8.4.2　电路特性随器件参数的变化

同理,可得

$$f_j^{min} = f_j(x_1^l, x_2^h)$$

其中

$$x_1^l = x_1^0 - t_1, \quad x_2^h = x_2^0 + t_2$$

因此,算法公式可写为

$$\begin{aligned} f_j^{max} &= f_j(x_i), \qquad j = 1, 2, \cdots, m \\ x_i &= x_i^0 + \operatorname{sgn}\frac{f_j}{x_i} t_i, \quad i = 1, 2, \cdots, n \\ f_j^{min} &= f_j(x_i) \qquad j = 1, 2, \cdots, m \\ x_i &= x_i^0 - \operatorname{sgn}\frac{f_j}{x_i} t_i, \quad i = 1, 2, \cdots, n \end{aligned} \tag{8.4.3}$$

按照(8.4.3)式,电路性能函数的最坏情况分析的计算步骤可归纳如下:

(1) 对器件参数的标称值进行电路模拟,并进行灵敏度分析,即计算 $f_j^0, j = 0, 1, \cdots, m$ 和 $f_j / x_i \big|_{x=x^0}, j = 1, 2, \cdots, m, i = 1, 2, \cdots, n$ 的数值。

(2) 对于性能函数 f_j,令 $x_i = x_i^0 + \operatorname{sgn}(f_j / x_i) t_i, i = 1, 2, \cdots, n$,在 $x = (x_1, x_2, \cdots, x_n)^T$ 处进行电路模拟,可以得出该性能函数的上限值 f_j^{max}。

(3) 对于性能函数 f_j,令 $x_i = x_i^0 - \operatorname{sgn}(f_j / x_i) t_i, i = 1, 2, \cdots, n$,在 $x = (x_1, x_2, \cdots x_n)^T$ 处进行电路模拟,可以得出该性能函数的下限值 f_j^{min}。

(4) 对所有性能函数 $f_j, j = 1, 2, \cdots, m$,重复上述过程(2)和(3)。

如果电路中器件参数的容差较小,则可以利用灵敏度分析的结果直接计算电路特性函数的最坏情况,公式(8.4.3)可以简化为

$$\begin{aligned} f_j^{max} &= f_j^0 + \sum_{i=1}^{n} \left| \frac{f_j}{x_i} \right| t_i \\ f_j^{min} &= f_j^0 - \sum_{i=1}^{n} \left| \frac{f_j}{x_i} \right| t_i \end{aligned} \qquad j = 1, 2, \cdots, m \tag{8.4.4}$$

一般意义上讲,按本节给出的最坏条件算法并不能保证求出电路性能函数的最坏情况。存在两种情况可能使最坏情况分析得出错误解,一种情况是最坏条件根本不出现在容差域的边界点上;另一种情况是最坏情况虽然出现在容差域的边界上,而上述算法的解却是错误的。图 8.4.3(a)中的函数 $f_k(x_i)$ 和(b)中的函数 $f_l(x_i)$ 分别示意画出了这两种情况。对于图(a)中函数 $f_k(x_i)$ 的情况,直接用灵敏度的信息不能得出电路性能函数的最坏解,对于图(b)中 $f_l(x_i)$ 的情况,这种直接采用灵敏度的方法得到的结果根本就是错误解。事实上的情况远比图中画出的情况复杂。

图 8.4.3　最坏条件发生的不同情况

(a) 最坏条件发生在容差域内部;　　(b) 最坏条件发生在容差域的边界

尽管如此,理论上可以证明,如果电路性能函数是其器件参数的拟凸(凹)函数,性能函数的最坏情况一定出现在容差域的边界点。如果电路的性能函数关于其器件参数连续,在参数容差不太大的条件下,即在器件参数标称值的一个较小的领域内,电路的性能函数都可以由拟凸(凹)函数很好地逼近。所以,如果器件参数的容差不大,采用本节给出的最坏情况分析方法进行最坏情况分析,可以得出相当精确的结果。

例 8.4.1　差分放大器如图 8.3.1 所示,两个晶体管的参数为 $\beta = 60$, $I_s = 1E-13A$,假设 5 个电阻的数值发生 5% 的变化,试用最坏情况分析估算节点电位 V(4)、V(5)、V(4) - V(5) 的最大值和最小值。

解　(1) 首先用电路仿真程序 PSpice 对电路进行直流工作点分析和直流灵敏度分析。灵敏度结果列于前面表 8.3.1 中。然后按照计算电路最坏情况的算法即式(8.4.3)求 V(4)、V(5) 和 V(4) - V(5) 的最大和最小值。计算结果列于表 8.4.1 中。

(2) 按式(8.4.4)所示的求电路最坏情况的算法,可得

$$V(4)^{max} = V(4)^0 + \sum_{i=1}^{5} \left| \frac{V(4)}{x_i} \right| t_i$$

$$= 7.8072 + (1.053E-3 \times 50) + (1.044E-3 \times 50) + (1.159E-3 \times 180) + (1.553E-3 \times 135) + (6.468E-11 \times 135)$$

$$= 7.8072 + 0.5231 = 8.3303$$

表 8.4.1　对图 8.3.1 所示的差分放大器进行最坏情况分析的结果

元件 / 输出变量		R_{B1}	R_{B2}	R_E	R_{C1}	R_{C2}
V(4)	f / x	> 0	< 0	> 0	< 0	< 0
	最大值点	1kΩ+ 50Ω	1kΩ- 50Ω	3.6kΩ+ 180Ω	2.7kΩ- 135Ω	2.7kΩ- 135Ω
	最小值点	1kΩ- 50Ω	1kΩ+ 50Ω	3.6kΩ- 180Ω	2.7kΩ+ 135Ω	2.7kΩ+ 135Ω
	计算结果	$V(4)^{normal}$ = 7.8072		$V(4)^{max}$ = 8.2982		$V(4)^{min}$ = 7.2481
V(5)	f / x	< 0	> 0	> 0	< 0	< 0
	最大值点	1kΩ- 50Ω	1kΩ+ 50Ω	3.6kΩ+ 180Ω	2.7kΩ- 135Ω	2.7kΩ- 135Ω
	最小值点	1kΩ+ 50Ω	1kΩ- 50Ω	3.6kΩ- 180Ω	2.7kΩ+ 135Ω	2.7kΩ+ 135Ω
	计算结果	$V(5)^{normal}$ = 7.8072		$V(5)^{max}$ = 8.2982		$V(5)^{min}$ = 7.2481
V(4) - V(5)	f / x	> 0	< 0	> 0	< 0	> 0
	最大值点	1kΩ+ 50Ω	1kΩ- 50Ω	3.6kΩ+ 180Ω	2.7kΩ- 135Ω	2.7kΩ+ 135Ω
	最小值点	1kΩ- 50Ω	1kΩ+ 50Ω	3.6kΩ- 180Ω	2.7kΩ+ 135Ω	2.7kΩ- 135Ω
	计算结果	$[V(4)-V(5)]^{normal}$ = 0.0　$[V(4)-V(5)]^{max}$ = 0.5943　$[V(4)-V(5)]^{min}$ = -0.6676				

$$V(4)^{min} = V(4)^0 - \sum_{i=1}^{5} \left| \frac{V(4)}{x_i} \right| t_i$$

$$= 7.8072 - 0.5231 = 7.2841$$

同理可得

$$V(5)^{max} = V(5)^0 + \sum_{i=1}^{5} \left| \frac{V(5)}{x_i} \right| t_i = 8.3303$$

$$V(5)^{min} = V(5)^0 - \sum_{i=1}^{5} \left| \frac{V(5)}{x_i} \right| t_i = 7.2841$$

$$[V(4)-V(5)]^{max} = [V(4)-V(5)]^0 + \sum_{i=1}^{5} \left| \frac{(V(4)-V(5))}{x_i} \right| t_i$$

$$= 0.0 + 0.6290 = 0.6290$$

$$[V(4)-V(5)]^{min} = 0.0 - 0.6290 = -0.6290$$

在实际电路中，全部器件参数全取最坏情况的可能性很小，所以如果在最坏情况下，电路性能仍能满足设计要求，则电路设计方案应该是很成功的方案，同时也是很保守的方案，对大批量生产的电子产品不一定是最经济有效的设计方案，只能作为参考。但对用于航天、反应堆等一些风险较大的设备中的电路，通常要求零失效率，其成本往往不是第一位的考虑因素，因而可能采用最坏情况设计的方案。

例 8.4.2　图 8.4.4 所示的是一个带通滤波器电路，用 PSpice 分析该滤波器输出的最坏情况。

图 8.4.4　带通滤波器

输入网单文件如下:

```
A Band Filter
R1  1  2  RMOD  10K
R2  2  0  RMOD  10K
C1  2  3  CMOD  6800P
C2  3  4  CMOD  6800P
R3  3  4  5.1K
XOP1  0  3  11  12  4  UA741
VDD  11  0  15
VEE  12  0  - 15
VI  1  0  AC  1     ; 图中为 EA
.MODEL  RMOD  RES (R= 1  DEV= 5%)
.MODEL  CMOD  CAP (C= 1  DEV= 10%)
.LIB E: \PSPICE4\LINEAR.LIB
.AC  DEC  10  10  100K
.WCASE  AC  VM(4)  YMAX  DEVICES  RC
.PROBE
.END
```

图 8.4.5　最坏情况分析的输出曲线

输入网单文件中,WCASE 是最坏情况分析语句,其中 AC VM(4)表示在交流小信号分析中对节点④输出电压幅度进行最坏情况分析; YMAX 表示计算相对于标称值情况的最大偏差; DEVICES RC 表示变化的元器件是电阻 R 和电容 C。

输出曲线如图 8.4.5 所示,其中曲线Ⅰ是标称值情况下的输出,曲线Ⅱ是最大偏差的曲线。

8.5 蒙特卡罗分析

蒙特卡罗方法是一种统计模拟方法。在电路 CAD 领域中,如果已知电路所有元器件参数的统计分布规律,根据这种分布规律随机地多次抽取元器件参数,并对随机抽取的电路进行计算机模拟,就可以估计出电路性能的统计分布规律。这就是电路的蒙特卡罗分析方法。蒙特卡罗方法的流程如图 8.5.1 所示。应用蒙特卡罗方法求解实际电路问题的过程,一般有如下几方面内容:

图 8.5.1 蒙特卡罗分析流程图

(1) 给出电路的拓扑结构、元器件参数以及元器件参数的统计分布规律,例如元器件参数的中心值和方差。

(2) 产生均匀分布的伪随机数抽样序列。

(3) 由均匀分布的伪随机抽样序列,建立电路元器件参数所指定分布的随机抽样序列。

(4) 用电路仿真工具对 N 组随机抽样的电路元器件参数所构成的电路依次进行模拟。

(5) 对模拟结果进行统计分析,计算出电路性能的统计规律:电路性能的中心值、方差以及电路的合格率等等。

8.5.1 随机数的产生

在计算机上实现蒙特卡罗分析,首先要产生电路元器件参数的随机抽样值,即在给定元器件参数的统计分布规律的条件下,在计算机上产生符合其分布规律的抽样值,这个过程称之为随机变量的模拟。最简单、最基本、最重要的随机变量是在[0,1]上均匀分布的随机变量。几乎所有其它分布的随机变量都可以由一个或多个均匀分布的随机变量变换得到。因此,如何产生均匀分布的随机变量的抽样值,是在计算机上实现蒙特卡罗分析的基础。

设 X 是[0,1]上均匀分布的随机变量,则其概率密度函数为

$$p(x)=\begin{cases}1 & 0\le x\le 1\\ 0 & \text{其它}\end{cases} \tag{8.5.1}$$

X 的数学期望是

$$E(X)=\int_0^1 x\,dx=\frac{1}{2} \tag{8.5.2}$$

其二阶矩 $E(X^2)=\frac{1}{3}\approx 0.33333$,方差 $\sigma^2(X)=E(X^2)-[E(X)]^2=1/12\approx 0.083333$。

产生均匀分布随机数的最通用方法是同余法。其算法公式为

$$x_k=\lambda x_{k-1}+c \pmod m \qquad k=1,2,\cdots \tag{8.5.3}$$

其中 x_k, λ, c 均为正整数。给定初值 x_0,就可由(8.5.3)式得到整数序列$\{x_k\}$。对每个 x_k 作变换 $u_k=x_k/m$,则$\{u_k\}$ $(k=1,2,\cdots)$就是[0,1]上的一个随机数序列。如果$\{u_k\}$通过了统计检验,那么就可以将 u_k 作为[0,1]上的均匀分布随机数。

在式(8.5.3)中,若 $c=0$,则称该算法为乘同余算法,并称 λ为乘子,x_0 为种子;若 $c\neq 0$,则称相应的算法为混合同余法。由(8.5.3)式产生的 $x_k(k=1,2,\cdots)$到一定长度以后,会出现周而复始的周期现象,即$\{x_k\}$可能由若干个子列的重复出现而构成。该重复出现的子列的最短长度,称为序列$\{u_k\}$的周期。显然,这种周期性与随机性是矛盾的,我们只能取$\{u_k\}$的一个周期做为可用的随机数序列。因此,为了产生足够多的随机数,$\{u_k\}$的周期应尽可能的大,这可以通过适当地选择式中 λ, c, x_0 等来实现。一般认为,当 $m=2^i$ 时,最大周期 $T=2^i-1$。所以通常称用数学方法产生的随机数为伪随机数。在实际应用中,只要这些伪随机数序列通过一系列的统计检验,就可以把它们当做‘真正’的随机数来使用。

我们把在计算机上实现的随机数序列也称为伪随机数序列。在许多计算机操作系统的 C 语言函数库中都提供了产生均匀分布随机数的库函数,这些库函数一般都通过了各种测试和检验,读者可以根据需要选用它们作为可用的均匀分布随机数。下面我们介绍几种 UNIX 操作系统中产生均匀分布随机数的库函数。

1. 库函数 rand( )和 srand( )

函数 rand( )用来产生 0 到 RAND_MAX 之间均匀分布的伪随机整数,RAND_MAX 是计算机的 C 编译器所确定的最大整数。函数 srand( )的作用是为 rand( )设定初值。

函数 rand( )采用的是乘同余法,其算法公式是

$$x_k = \lambda x_{k-1} \pmod{RAND\_MAX} \tag{8.5.4}$$

适当地选择乘子 λ 和 RAND_MAX, 可以得到周期长、统计特性好的伪随机数序列。例如选择

$$\begin{aligned} \lambda &= 7^5 = 16807 \\ RAND\_MAX &= 2^{31} - 1 = 2147483647 \end{aligned} \tag{8.5.5}$$

就很不错。但要特别注意的是: 在 32 位计算机上, λx_{k-1} 有可能大于 2^{32}, 而出现溢出现象。

一般情况下, 常用的是 0 到 1 间的均匀分布随机数, 此时需要将上面得到的随机整数除以 RAND_MAX, 即

$$x = rand()/(RAND\_MAX + 1.0) \tag{8.5.6}$$

2. 库函数 random( ) 和 srandom

函数 random( ) 和 srandom 是 UNIX 系统提供的一对较好的伪随机数发生器。对于 32 位的 C 编译器, random( ) 产生 0 至 $2^{31}-1$ 间的均匀分布整数。它采用了非线性"加反馈"的随机数产生算法, 并利用了由 31 个长整数组成的隐含状态表。该算法产生的随机数的最大周期长达 $16\times(2^{31}-1)$, 比函数 rand( ) 的周期要长得多。而且, rand( ) 产生的伪随机数序列中每个整数的几个低位的随机性较差, 而 random( ) 产生的伪随机数每位都可用。还有一个不同之处是, rand( ) 的初始化只需一个初值, 通过调用函数 srand( ) 完成; 而 random( ) 的初始化则要提供 256 字节长的状态表来完成。

3. 库函数 drand48( ) 和 srand48( )

函数 drand48( ) 的作用是产生[0, 1]之间均匀分布的随机数。它采用了线性同余法和 48 位整数运算来产生伪随机序列, 算法公式为

$$x_k = ax_{k-1} + c \pmod{m} \tag{8.5.7}$$

其中

$$m = 2^{48}, \quad a = 5DEECE66D = 273673163155, \quad c = B16 = 138$$

函数 drand48( ) 首先根据式(8.5.7)的算法产生出 48 位字长的伪随机整数 x_k, 将这个伪随机整数存在内部缓冲区内为产生下一个 48 位随机整数使用, 然后复制 x_k 的高 32 位得到一个 32 位的伪随机整数, 最后将这个 32 位的伪随机整数变换到[0, 1]之间。用函数 srand48( ) 对伪随机数发生器 drand48( ) 进行初始化, 只对 48 位整数的高 32 位进行了初始化, 而其低 16 位被设定为随机值。显然, 这是一种统计特性比较好的伪随机数发生器。

在电路的蒙特卡罗分析中, 一般电路的元器件参数的数目较多, 蒙特卡罗分析的抽样点数也较大, 因此要求随机数发生器的周期足够长, 而且随机性要好, 否则就会影响蒙特卡罗分析的精度。对于不同的元器件参数, 要得到不同的伪随机数序列, 则要依靠设定初值 x_0 来实现, 库函数 srand( ) 的作用就是为 rand( ) 设置初值。还应该特别注意的是, 在使用所产生的伪随机数前, 必须对它们进行统计检验, 包括参数检验、均匀性检验、独立性检验和组合规律性检验, 以确定这些随机数是否符合均匀性、独立性、周期长等均匀分布随机数所应满足的统计特性。

8.5.2 随机变量的抽样

获得均匀分布的随机数之后,可以对它们进行变换,以得到其它统计分布规律的随机数,因此由均匀随机数到给定分布的随机数的各种变换方法,就称为该分布的各种抽样方法。

1. 变换法

假定随机变量 X 在[0, 1]区间服从均匀分布,其概率密度函数为

$$p(x) = \begin{cases} 1 & 0 \le x \le 1 \\ 0 & \text{其它} \end{cases} \tag{8.5.8}$$

用式

$$y = \phi(x) \tag{8.5.9}$$

将 X 变换为随机变量 Y。若 Y 的概率密度函数是 p(y),那么根据概率的基本定理,可得

$$\int p(y)\,dy = \int p(x)\,dx \tag{8.5.10}$$

即

$$p(y) = p(x)\left|\frac{dx}{dy}\right| = p[\psi(y)]\left|\frac{dx}{dy}\right| \tag{8.5.11}$$

其中

$$\psi(y) = \phi^{-1}(y) = x$$

即 $\psi(y)$ 为 $\phi(x)$ 的反函数,随机抽样中经常采用的反变换法,是变换法(8.5.11)式的特殊情况。也就是说,若已有[0,1]区间均匀随机变量 X,需要产生分布函数为 F(y) 的连续随机变量 Y,则产生 X 的反变换公式为

$$F(y) = x \tag{8.5.12}$$

即

$$y = F^{-1}(x) \tag{8.5.13}$$

式(8.5.13)的直观示意图如图 8.5.2 所示。若给定[0,1]区间均匀变量 X 的一个数值 x_i,则 $y_i = F^{-1}(x_i)$ 即为分布函数为 F(y) 的连续随机变量 X 的一个数值。

图 8.5.2 反变换法的示意图

下面是两个利用这种反变换法将均匀分布随机变量变换为其它分布随机变量的抽样例子。

(1) 指数分布的抽样

随机变量 Y 服从指数分布, 其概率密度函数为

$$p(y)=\begin{cases}\lambda e^{-\lambda y} & y>0\\ 0 & \text{其它}\end{cases} \tag{8.5.14}$$

其中 $\lambda>0$ 为常数, 对应的概率分布函数是

$$F(y)=\begin{cases}1-e^{-\lambda y} & y>0\\ 0 & y\le 0\end{cases}$$

根据(8.5.12)式, 在[0,1]上均匀分布的随机变量 X 与指数分布的随机变量 Y 之间有关系式 $x=F(y)$, 即它们之间的变换关系为

$$y=F^{-1}(x)=\begin{cases}-\dfrac{\ln(1-x)}{\lambda} & 0\le x<1\\ 0 & \text{其它}\end{cases} \tag{8.5.15}$$

如果随机变量 X 在区间[0,1]上均匀分布, 则随机变量 1-X 也在区间[0,1]上均匀分布, 所以式(8.5.15)等效于

$$y=F^{-1}(x)=\begin{cases}-\dfrac{\ln(x)}{\lambda} & 0\le x<1\\ 0 & \text{其它}\end{cases} \tag{8.5.16}$$

如果要求产生指数分布的随机变量的抽样值, 首先产生在区间[0,1]均匀分布的随机变量抽样值, 按式(8.5.16)变换后就可得到指数分布随机变量的抽样值。

(2) 标准正态分布的抽样

设 x_1, x_2 是在区间[0,1]上两个相互独立的均匀分布的随机变量, 作如下变换:

$$\begin{aligned}y_1&=\sqrt{-2\ln x_1}\cos(2\pi x_2)\\ y_2&=\sqrt{-2\ln x_1}\sin(2\pi x_2)\end{aligned} \tag{8.5.17}$$

解此方程组, 得到反变换:

$$\begin{aligned}x_1&=\exp\left(-\frac{1}{2}(y_1^2+y_2^2)\right)\\ x_2&=\frac{1}{2\pi}\arctan\frac{y_2}{y_1}\end{aligned} \tag{8.5.18}$$

可以推导出随机变量 y_1, y_2 的二维联合概率密度函数为:

$$\begin{aligned}p(y_1,y_2)&=p(x_1,x_2)\,\|\grave{e}_y X\|=\frac{1}{2\pi}\exp\left(-\frac{1}{2}(y_1^2+y_2^2)\right)\\ &=\frac{1}{2\pi}\exp\left(-\frac{y_1^2}{2}\right)\exp\left(-\frac{y_2^2}{2}\right)\end{aligned} \tag{8.5.19}$$

其中, $\grave{e}_y X$ 为雅可比(Jacobian)矩阵, $\|\cdot\|$ 为行列式的绝对值。由此可知, y_1, y_2 为相互独立且服从标准正态分布 N(0,1) 的随机变量。

反变换法也可以用来产生离散分布的随机变量 Y。设 Y 的可能取值为 $y_1, y_2, \cdots, y_k, \cdots$, 其密度函数为

$$p_k=P(Y=y_k),\quad k=1,2,3,\cdots \tag{8.5.20}$$

其分布函数为

$$F(y) = P(Y \leq y) = \sum p_k \quad y \geq y_k \tag{8.5.21}$$

先产生[0,1]区间均匀分布随机变量 X,再计算满足

$$F(y_{k-1}) < x \leq F(y_k) \tag{8.5.22}$$

的 k 值,则 y_k 即为所需随机数。图 8.5.3 是用反变换产生离散分布随机数的示意图,它与图 8.5.2 是相似的。

图 8.5.3　产生离散分布随机数的反变换法

例 8.5.1　离散随机变量 Y 的统计分布为表 8.2.2 所示的 100Ω电阻的离散分布,用反变换法由[0,1]区间均匀分布的随机变量 X 产生离散正态分布的随机变量 Y 的抽样。

解　Y 的分布函数 $F(y) = P\{Y \leq y\}$ 为

$$F(y) = \begin{cases} \sum_{i \leq k} p_i & y_k < y \leq y_{k+1};\ k = 1, 2, \cdots, 9 \\ 1 & y > y_{10} \\ 0 & y \leq y_1 \end{cases} \tag{8.5.23}$$

F(y)的数值如表 8.5.1 所示。如果抽样得到均匀分布随机变量 X 的值在 0～0.022 之间,取离散随机变量 Y 的值为 95;如果抽样得到均匀分布随机变量 X 的值在 0.022～0.072 之间,取离散随机变量 Y 的值为 96;以此类推,即如果抽样得到均匀分布随机变量 X 的值在 $F(y_k)$～$F(y_{k+1})$之间,取离散随机变量 Y 的值为 y_{k+1}。

表 8.5.1　离散随机变量的概率分布函数

区间 i	1	2	3	4	5	6	7	8	9	10
y_i	95	96	97	98	99	100	101	102	103	104
p_i	0.022	0.050	0.101	0.161	0.192	0.18	0.141	0.111	0.032	0.010
y	95～96	96～97	97～98	98～99	99～100	100～101	101～102	102～103	103～104	104～105
F(y)	0.022	0.072	0.173	0.334	0.526	0.706	0.847	0.958	0.990	1.000

在对电路进行蒙特卡罗分析时,如果器件参数的统计分布规律由直方图给出,则采用上述变换方法是离散随机变量模拟的最基本方法。

2. 舍选抽样法

假定随机变量 X 的概率密度函数 f(x) 有界，其取值区间为[a, b]，即有

$$c = \max\{f(x) \mid a \le x \le b\} \tag{8.5.24}$$

舍选抽样法产生这种分布的随机数的步骤如下：

(1) 产生[a, b]区间均匀分布随机数 x；

(2) 产生[0, c]区间均匀分布随机数 y；

(3) 当 $y \le f(x)$ 时，接受 x 为所需分布的随机数；否则重复步骤(1)～(3)。

舍选抽样的示意图如图 8.5.4 所示，在图示矩形内任投一点 $t_i(x_i, y_i)$，若 t_i 在曲线 f(x) 之下，则表示 x_i 即为所求的随机数(如 t_1)；否则(如 t_2)，重新进行投点试验。

图 8.5.4　舍选抽样法示意图

用舍选抽样法产生标准正态分布随机数时，标准正态分布的密度函数为

$$f(x) = \frac{1}{\sqrt{2\pi}} e^{-x^2/2}$$

产生[-a, a]区间标准正态分布随机数的步骤如下：

(1) 产生[-a, a]区间均匀分布随机数 x；

(2) 产生 $\left[0, \frac{1}{\sqrt{2\pi}}\right]$ 区间均匀分布随机数 y；

(3) 当 $y \le \frac{1}{\sqrt{2\pi}} e^{-x^2/2}$ 时，接受 x 为标准正态分布随机数；否则，重复步骤(1)～(3)。

8.5.3　蒙特卡罗法在电路仿真中的应用

一旦解决了在计算机上实现伪随机数和电路器件参数的抽样问题以后，蒙特卡罗问题就可归结为在不同的器件参数抽样序列情况下重复进行电路仿真，并对仿真结果进行统计分析，最后求出电路性能的数学期望、均方差以及电路的合格率、合格率标准偏差等统计特性。

目前许多商用的电路仿真器都具有蒙特卡罗分析功能，如 PSpice，HSPICE 及 Mentor Graphics，Cadence，Viewlogic 等 EAD 公司的电路仿真工具。用这些仿真工具进行蒙特卡罗分析时，可以输出每次抽样时电路仿真的结果，以及经过用户指定的 N 次抽样仿真后计算出的电路性能的均值、最大值、最小值、方差和电路性能的统计分布直方图。

如果用户指定电路性能的约束条件,还能计算出电路的合格率及合格率的偏差。

例 8.5.2　对前面图 8.4.4 所示的带通滤波器电路进行蒙特卡罗分析。

需在输入网单文件中设置蒙特卡罗分析语句,即

```
.MC 10 AC VM(4) YMAX OUTPUT ALL
```

其中,.MC 是蒙特卡罗分析语句描述关键字; 10 表示分析计算 10 次,第 1 次进行标称值分析,后 9 次进行蒙特卡罗分析; AC VM(4) 表示是在交流小信号分析中对节点④的输出电压幅度进行蒙特卡罗分析; YMAX 表示求出每个蒙特卡罗输出与标称值的最大差值; OUTPUT ALL 表示 10 次计算结果都输出。

该滤波器蒙特卡罗输出结果波形如图 8.5.5 所示。

图 8.5.5　蒙特卡罗分析的输出波形

例 8.5.3　对图 8.5.6 所示 OTL 电路进行蒙特卡罗分析,电路各元件容差是 5%。在电路响应(中点电压 V(10)) 容差范围是 6V±0.1V 的情况下,计算电路的合格率与标准偏差。

图 8.5.6　OTL 电路

解　(1) 由元件标称值计算 OTL 电路的直流工作点如下:

(1)	7.6496	(2)	7.0525	(3)	11.2758	(5)	0.0000
(8)	5.3178	(9)	0.1156	(10)	5.9756	(11)	6.0246
(12)	12.0000	(13)	5.9745	(14)	5.9756	(20)	0.0057

其中,中点电压

$$V(10) = 5.9756V$$

(2) 进行蒙特卡罗分析结果如下:

合格率: 88%

合格率标准偏差: 2.298%

中点电压(V(10))均值: 5.953V

中点电压标准偏差: 0.164V

中点电压直方图如图 8.5.7 所示。

图 8.5.7　OTL 电路中点电压的统计直方图

蒙特卡罗分析是"真实"模拟电路实际制造情况的一种有效方法,电路的性能函数的性质没有作任何近似,可以在静态、时域和频域分析器件参数的随机性对电路性能的影响。但是,由于蒙特卡罗分析要多次进行电路模拟,其计算成本很高,对计算机存储资源的要求也很高。特别是时域蒙特卡罗分析和频域蒙特卡罗分析,对于每一个抽样电路,要在每一个时刻(时域抽样点)或每一个频率(频域抽样点)进行电路模拟,并存储模拟结果,计算时间和存储量都可能很大。另外,如果电路模型不准确或器件参数的统计分布规律不准确,都会影响蒙特卡罗分析的准确性。在进行蒙特卡罗分析时,总抽样点数 N 的选取要进行权衡,抽样点数越大,统计分析结果越准确,但随之而来的是计算成本的上升。

习　　题

8.1　对题图 8.1 所示电路,

题图　8.1

(1) 采用伴随网络法,求 V(2) 对所有元件参数的灵敏度值。

(2) 采用导数网络法,计算输出向量对 R_2, E_1 和 I_1 的灵敏度值。

(3) 若所有电路元件参数变化+ 5%,求 V(2) 的变化。

(4) 若所有电路元件参数容差为±5%,求最坏情况 $V(2)^{max}$ 和 $V(2)^{min}$。

8.2　对题图 8.2 所示各电路,

题图　8.2

(1) 若所有电路元件参数变化+ 10%,求 V(2) 的变化。

(2) 若所有电路元件参数容差为±5%,求最坏情况 $V(2)^{max}$ 和 $V(2)^{min}$。

8.3 在角频率为 w 时,求题图 8.3 所示电路中,

题图 8.3

(1) 电容和电感变化+ 10% 时,V(2) 的变化。

(2) 电容和电感容差为±10% 时,最坏情况 $V(2)^{max}$ 和 $V(2)^{min}$。

第 9 章　大规模电路的仿真技术

9.1　引　　言

近年来,由于集成电路技术的高速发展,在一个芯片上可以集成数万乃至数十万、数百万个晶体管的电路,因而所需进行电路模拟的电路规模也十分庞大,用我们前面介绍的求解线性代数方程、非线性代数方程和非线性常微分方程的方法来完成如此大规模电路的电路模拟问题,在计算时间和存储容量上都无法接受。为了降低大规模集成电路的模拟时间和存储要求,人们根据大规模电路的特点提出了一些适用于大规模电路的电路模拟技术和算法,这些技术和算法大致可以分为下面几个方面:

1. 宏模型技术

宏模型技术是一种简化电路模型的技术,我们在第 3 章 3.5 节宏模型中曾做了介绍。它是将大规模电路系统中的集成电路芯片或子模块电路等效为简化模型或端口特性模型,称之为宏模型。例如,运放、相乘器、比较器、开关电源电路以及数字电路中的单元电路等等。宏模型在电性能上应当能够满足原电路的精度,结构上却比原电路要简单得多,使电路的元件数和节点数大大减小,电路方程的规模也随之降低。显然,这对于减少方程的求解时间和存储空间是十分有利的。

目前商用的电路仿真软件上已备有宏模型库和宏模型的自动生成工具,有不少电路设计人员在从事宏模型的研究工作,宏模型的种类和数量都在不断地增加。这不仅加速了大规模电路的仿真时间,而且为电路的系统级模拟和电路综合优化奠定了基础。

2. 电路的分解技术

电路的分解技术是依据大规模电路的特点,将大规模电路的模拟问题分解为若干子电路或模块的模拟问题。这种分解技术,可以是电路结构级的分解,也可以是电路功能上的分解,或者是算法(电路方程)级的分解;当然,也可能是几种分解方法的混合应用。

(1) 撕裂技术

撕裂技术是按照大规模电路是由若干子电路组成的特点,将子电路之间的连接支路(或节点)撕裂开,使整个电路分解为若干分离的小电路,通过对这些小电路的求解而获得整个电路的解。采用撕裂技术所形成的电路方程组形式上比较特殊,因而计算量要远远小于通常的电路方程求解。这实质上是一种按照电路结构进行分解的方法。

(2) 潜伏技术

潜伏是指一些大规模电路在模拟时间区间内的某段时间里,电路中一些子电路的电压、电流变化甚小,可视为常量,我们称这些子电路在这段时间内处于潜伏状态。如果我们能判断出某个子电路在当前时刻处于潜伏状态,就不必计算该子电路的电压、电流在此时刻的值,而用前面时刻的值代替,从而可以节省计算时间。在大规模数字电路的模拟过程中,往往在大量时间内,有大量子电路处于潜伏状态,充分利用这种潜伏特性可以使计算

时间大大地节省。因此这种方法必然要对大电路进行结构级或算法级的划分。

(3) 松弛技术

松弛技术属于算法级的分解技术,它着眼于如何有效地求解电路方程。松弛技术是以经典的求解方程的松弛型算法为基础,研究电路方程的非直接解法。前面所介绍的解方程方法,例如高斯消去法,在求解电路方程 TX= B 时,是将 TX= B 作为一个联立方程来处理的,即方程组 TX= B 中的 n 个方程是紧密相关的,或者说是紧耦合的; 只能联立求解,而不能一个一个单独求解。而松弛技术却是将紧耦合的 n 个方程去耦(或者说松弛),把联立求解方程变为一个一个地求解方程。这样,n 个方程可能同时求解,或作并行处理; 也可以更有效地利用潜伏技术。松弛技术特别适合于分析大规模 MOS 数字集成电路。

采用上述分解技术可以将大电路分解为若干子电路求解,从而减少计算量,此外,还有一个显著的优势是便于进行并行处理,即采用并行算法或并行计算机同时处理多个子电路,以缩短计算时间。

3. “软硬兼施”技术

集成电路芯片与其 CAD 设计工具之间的关系,始终是相辅相成、相互促进、共同发展的。近年来随着集成技术的迅速发展,集成电路的集成度飞速增加,复杂程度越来越大,因而在集成电路的整个设计过程和每个设计环节都必须使用 CAD 技术和工具,并且迫切地对 CAD 技术的可靠性、精度和自动化提出更高的要求。同时 CAD 工具的发展对提高集成电路的设计效率,缩短设计周期等又起着至关重要的作用。但总的趋势是硬件发展要快于软件的发展,因此在 CAD 软件工具的开发中,将软件与硬件有机地结合起来,“软硬兼施”,是当前 CAD 技术发展的一种新的趋势。

(1) 在 CAD 软件工具中加硬件仿真器

当新的集成电路芯片设计出来后,为了仿真由它组成的电路系统,将芯片插在硬件仿真器上,通过 A/ D 和 D/ A 接口,将芯片输出特性转换为 CAD 工具所能接受的数据,参加软件级的整体系统仿真。目前几个著名的 EDA 公司 Mentor Graphics 和 Cadence 等都有这种硬软件组合的仿真功能,它为大规模电路的系统仿真开辟了新天地。

(2) 晶体管表格模型硬件化

在电路仿真中计算晶体管的非线性数学模型以求得晶体管的端电压和端电流,几乎占据了整个电路仿真的 80% 以上的时间。我们将晶体管端电流与端电压之间以及它们与模型参数之间的函数关系制成表格,在电路仿真中,查表要比计算函数值快得多,而且能够保证模型的精度。例如,MOS 晶体管的源漏电流与端电压的关系 $I_{DS}= f(U_{DS}, U_{GS}, U_{SB})$,就可以制成一个包含各个电压范围内的全部 I_{DS} 值的表格。使用表格模型需要的存储比使用数学模型要多得多,目前计算机的内存和外存都相当大,完全可以满足表格模型对计算机内外存的需求。也有的 EDA 公司将表格模型硬件化,烧在 ROM 中,这样既可以节省存储,数据的读取也更加方便。

为了使大家对大规模电路的模拟方法有更加深入的了解,我们在这一章中着重介绍其中几个主要算法: 撕裂法、波形松弛法、多级牛顿算法和目前应用广泛的混合模拟算法。

9.2 撕 裂 法

大规模电路往往是由大量相同的子电路构成的,如各种门电路、触发器、寄存器、运放等等,还可能是专门完成某种功能的功能块。这些子电路之间可能只有少量的支路相连接,若将连接支路断开(撕裂),则整个电路变为若干个互不相连的子电路。撕裂法的基本思想就是通过对电路的支路和节点进行适当的划分,把大电路分解为若干小电路,通过对小电路的求解而获得整个电路的解。采用撕裂法所形成的电路方程的系数矩阵具有加边块对角形式,非零元素量小,所以在存储要求和计算量上都会大为减少。

常用的撕裂法有支路撕裂法和节点撕裂法。

9.2.1 支路撕裂法

在支路撕裂法中,一个大规模电路是由多个子电路 $s_1, s_2, \cdots, s_k$ 组成的,子电路之间通过少量支路 $l_1, l_2, \cdots, l_m$ 连接,如图 9.2.1 所示。如果将这些连接支路断开(撕裂),那么整个电路就变为 k 个互不相连的子电路。因此称连接支路为撕裂支路,其余各子电路中的支路称为剩余支路。所以,整个电路的支路分为两部分,一部分是撕裂支路,另一部分是剩余支路,分别用 α 和 $\bar{\alpha}$ 表示。那么,整个电路的关联矩阵 A 也相应由两个子阵 A_l 和 A_s 组成,即

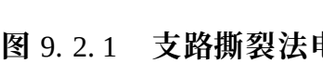


图 9.2.1 支路撕裂法电路结构示意图

$$A = [\underset{\bar{\alpha}}{A_s} \quad \underset{\alpha}{A_l}] \tag{9.2.1}$$

这样整个电路的 KCL、KVL 和支路特性方程可以表示为如下形式。

(1) KCL

$$AI = 0$$

即

$$[A_s \quad A_l] \begin{matrix} I_s \\ I_l \end{matrix} = 0 \tag{9.2.2}$$

其中,I_s 和 I_l 分别表示剩余支路和撕裂支路电流向量。或者将上式写为

$$A_s I_s + A_l I_l = 0 \tag{9.2.3}$$

(2) KVL

$$U_b = A^T V_n$$

将(9.2.1)式代入上式,得

$$U_s = A_s^T V_n \tag{9.2.4}$$

$$U_l = A_l^T V_n \tag{9.2.5}$$

其中,$U_b = [U_s \quad U_l]^T$, U_s, U_l 分别表示剩余支路和撕裂支路电压向量。

(3) 支路特性方程

撕裂支路:

$$U_l = Z_l I_l + Z_l I_{Jl} - E_l \tag{9.2.6}$$

剩余支路:

$$I_s = Y_s V_s + Y_s E_s - I_{Js} \tag{9.2.7}$$

其中,Z_l 为撕裂支路阻抗矩阵,Y_s 为剩余支路导纳矩阵,E_l,E_s 分别为撕裂支路与剩余支路的电压源向量,I_{Jl}、I_{Js}分别为它们的电流源向量。

如果我们以节点电位 V_n 和撕裂支路电流 I_l 为变量来列电路方程,将(9.2.4)式代入(9.2.7)式得

$$I_s = Y_s A_s^T V_n + Y_s E_s - I_{Js} \tag{9.2.8}$$

再将(9.2.8)式代入(9.2.3)式,则有

$$A_s Y_s A_s^T V_n + A_l I_l = I_{Js0} \tag{9.2.9}$$

其中

$$I_{Js0} = A_s I_{Js} - A_s Y_s E_s \tag{9.2.10}$$

同样,将(9.2.5)式代入(9.2.6)式,得

$$A_l^T V_n - Z_l I_l = E_{l0} \tag{9.2.11}$$

其中

$$E_{l0} = -E_l + Z_l I_{Jl} \tag{9.2.12}$$

把(9.2.9)式和(9.2.11)式联立在一起,写成矩阵方程形式:

$$\begin{matrix} A_s Y_s A_s^T & A_l \\ A_l^T & -Z_l \end{matrix} \begin{matrix} V_n \\ I_l \end{matrix} = \begin{matrix} I_{Js0} \\ E_{l0} \end{matrix} \tag{9.2.13}$$

这就是由支路撕裂法得到的电路方程。方程系数矩阵中的左上角阵 $A_s Y_s A_s^T$ 是由剩余支路构成的节点导纳矩阵。可以很容易地用节点法,由各剩余支路对节点导纳方程组的贡献来形成。系数矩阵中其它三个子阵 A_l,A_l^T,Z_l 都是只与撕裂支路有关的矩阵。在右端向量中,I_{Js0}是仅与剩余支路有关的等效电流源向量,E_{l0}是仅与撕裂支路有关的等效电压源向量。

假定从原电路中移去撕裂支路后,k 个子电路 s_1,s_2,…,s_k 是相互独立,不相连的,而且整个电路的节点全都分布在这 k 个子电路中,不存在仅由撕裂支路连接的节点。我们把整个电路中的 n 个独立节点分为 k 个子集 n_1,n_2,…,n_k,每个子集分别对应一个子电路所包含的独立节点,那么节点电位向量 V_n 和等效电流源向量 I_{Js0}可以表示为如下形式:

$$V_n = \begin{matrix} V_{n1} \\ V_{n2} \\ \vdots \\ V_{nk} \end{matrix} \qquad I_{Js0} = \begin{matrix} I_{Js01} \\ I_{Js02} \\ \vdots \\ I_{Js0k} \end{matrix}$$

关联矩阵 A_s 也可以相应分块为

$$A_s = \begin{matrix} & s_1 & s_2 & \cdots & s_k \\ n_1 & A_{s1} & & & \\ n_2 & & A_{s2} & & \\ \vdots & & & W & \\ n_k & & & & A_{sk} \end{matrix}$$

因此 $A_s Y_s A_s^T$ 可以分块表示为

$$A_s Y_s A_s^T = \begin{matrix} A_{s1}Y_{s1}A_{s1}^T & & & \\ & A_{s2}Y_{s2}A_{s2}^T & & \\ & & \mathbf{W} & \\ & & & A_{sk}Y_{sk}A_{sk}^T \end{matrix} = \begin{matrix} Y_{n1} & & & \\ & Y_{n2} & & \\ & & \mathbf{W} & \\ & & & Y_{nk} \end{matrix} \tag{9.2.14}$$

其中

$$Y_{ni} = A_{si}Y_{si}A_{si}^T \qquad i = 1, 2, \cdots, k$$

撕裂支路的关联矩阵也可按与子电路的连接关系作如下分块:

$$A_l = [A_{l1} \quad A_{l2} \quad \cdots \quad A_{lk}]$$

于是电路方程(9.2.13)**可以写为**

$$\begin{matrix} Y_{n1} & & & A_{l1} & V_{n1} & I_{Js01} \\ & Y_{n2} & & A_{l2} & V_{n2} & I_{Js02} \\ & & \mathbf{W} & \vdots & \vdots = & \vdots \\ & & Y_{nk} & A_{lk} & V_{nk} & I_{Js0k} \\ A_{l1}^T & A_{l2}^T & \cdots & A_{lk}^T & -Z_l & I_l \end{matrix} \quad \begin{matrix} \\ \\ \\ \\ E_{l0} \end{matrix} \tag{9.2.15}$$

这就是我们要求的支路撕裂方程组,它的系数矩阵是一个具有加边的块对角矩阵,未知变量是节点电压和撕裂支路电流。如果电路的独立节点数为 n,**撕裂支路数为** m,**那么**(9.2.15)**式所表示的撕裂方程组的维数为**(n+ m),**系数矩阵的加边宽度为** m。

具有加边的块对角矩阵实际上是一种结构特殊的稀疏矩阵,它的求解比满阵求解的运算量要小得多。可以看出,式(9.2.15)**的主要运算量是对每个子电路的节点导纳子阵** Y_{ni}**进行** LU **分解和对撕裂支路阻抗矩阵** Z_l **进行** LU **分解。若整个电路含有** k **个子电路,** m **个撕裂支路,每个电路内部有** p **个独立节点,那么分解每个** Y_{ni}**需要约** $k\times\frac{1}{3}p^3$ **次乘除运算量,分解** Z_l **需要约** $\frac{1}{3}m^3$ **次乘除运算量,因此整个电路的运算量** $N_t=\frac{1}{3}(kp^3+m^3)$**。如果将整个电路作为一个整体,方程组的维数是** kp**,** LU **分解所需乘除运算量** $N_n=\frac{1}{3}(kp)^3$**。例如** $k=10, p=10, m=2k=20$**,则** $N_t=\frac{1}{3}(kp^3+m^3)=6\times10^3$, $N_n=\frac{1}{3}(kp)^3\approx3\times10^5$**。由此可见,当电路中子电路数越大,撕裂支路数越小时,这种支路撕裂法的运算量减小得越加显著。**

9.2.2 节点撕裂法

节点撕裂法的特点是它仅以节点电位作为电路方程的未知变量,构成的电路方程是

节点导纳方程组。它的系数矩阵同样是具有加边的块对角矩阵。

我们以图 9.2.2 所示的节点撕裂电路结构示意图为例，说明节点撕裂法的算法特点。

(1) 整个电路由 k 个子电路 $s_1, s_2, \cdots, s_k$ 组成。设 n 为电路中独立节点的集合，将它划分为两个子集 n_1 和 n_2。n_1 为各子电路内部的节点集合，即 $n_1^1, n_1^2, \cdots, n_1^k$ 分别属于子电路 $s_1, s_2, \cdots, s_k$ 的节点集合，$n_1 = \{n_1^1, n_1^2, \cdots, n_1^k\}$。$n_2$ 为撕裂节点集合。

(2) 设 α 为电路中所有支路的集合，将它划分为两个子集 α 和 α。仅与撕裂节点相连的支路属于子集 α，如图 9.2.2 中连接 n_{m1} 与 n_{m2} 和连接 n_{m1} 和 n_{m3} 的支路。其它支路属于子集 α，$\alpha = \{\alpha^1, \alpha^2, \cdots, \alpha^k\}$，其中 α^i 是子电路 s_i 内部支路及连向 s_i 的子电路外部的支路集合。

图 9.2.2　节点撕裂法电路结构示意图

按照上述划分原则所确定的电路关联矩阵具有如下形式：

$$A = \begin{array}{c|ccccc} & \alpha_1^1 & \alpha_1^2 & \cdots & \alpha_1^k & \alpha_2 \\ \hline n_1^1 & A_{11}^1 & & & & \\ n_1^2 & & A_{11}^2 & & & \\ \vdots & & & W & & 0 \\ n_1^k & & & & A_{11}^k & \\ \hline n_2 & A_{21}^1 & A_{21}^2 & \cdots & A_{21}^k & A_{22} \end{array} \tag{9.2.16}$$

其中 A_{11}^i 描述了 α^i 中的支路与 n_1^i 中节点的关联关系，A_{21}^i 描述了 α^i 中的支路与 n_2 中撕裂节点的关联关系。由于各子电路之间无支路相连，且 α 中的支路不与 n_1 中节点相连，故 A 矩阵最右端的那一列相应的子阵为零。

同理，整个电路的支路导纳矩阵形式为

$$Y = \begin{array}{c|ccccc} & \alpha_1^1 & \alpha_1^2 & \cdots & \alpha_1^k & \alpha_2 \\ \hline \alpha_1^1 & y_{11}^1 & & & & y_{12}^1 \\ \alpha_1^2 & & y_{11}^2 & & & y_{12}^2 \\ \vdots & & & W & & \vdots \\ \alpha_1^k & & & & y_{11}^k & y_{12}^k \\ \hline \alpha & y_{21}^1 & y_{21}^2 & \cdots & y_{21}^k & y_{22} \end{array} \tag{9.2.17}$$

按照节点法，电路的节点导纳方程为

$$AYA^T V_n = A(I_S - YE) \tag{9.2.18}$$

或写为

$$Y_n V_n = I_{Jn} \tag{9.2.19}$$

其中, $Y_n = AYA^T$, $I_{Jn} = A(I_S - YE)$, I_S, E 分别为电流源和电压源向量。

将(9.2.16), (9.2.17)式代入(9.2.19)式, 则有

$$\left[\begin{array}{cccc|cc} Y_{11}^1 & & & & Y_{12}^1 & V_{n1}^1 \\ & Y_{11}^2 & & & Y_{12}^2 & V_{n1}^2 \\ & & W & & \vdots & \vdots \\ & & & Y_{11}^k & Y_{12}^k & V_{n1}^k \\ \hline Y_{21}^1 & Y_{21}^2 & \cdots & Y_{21}^k & Y_{22} & V_{n2} \end{array}\right] = \begin{array}{c} I_{Jn1}^1 \\ I_{Jn1}^2 \\ \vdots \\ I_{Jn1}^k \\ \hline I_{Jn2} \end{array} \tag{9.2.20}$$

这就是节点撕裂法的电路方程组, 它的系数矩阵也是一个具有加边的块对角矩阵, 是节点导纳矩阵, 未知变量是节点电位。如果电路的独立节点数为 n, 撕裂节点数为 n_2, 那么(9.2.20)式所表示的节点撕裂方程组是 n 维的, 加边宽度为撕裂节点数 n_2。与支路撕裂法相同, 如果电路划分的子电路数越多, 撕裂节点数越少, 那么节点撕裂方程组求解所需的运算量就越小。

9.3 松弛技术

前面介绍的撕裂法的着眼点在于如何利用大规模电路的特点构造一个便于直接求解的电路方程组, 本节所讨论的松弛技术则有所不同, 它是以经典的求解方程的松弛算法为基础, 结合大规模电路(主要是 MOS 数字集成电路)的特点, 实现电路方程组的非直接解法, 研究如何有效地求解电路方程组, 从而减少运算量、存储量, 并提高收敛速度。

电路模拟技术中, 三种基本的分析领域是: 直流、交流(频域)和瞬态(时域)分析。这三种分析所需求解的电路方程类型也不相同。对直流分析, 所需求解的电路方程是线性或非线性方程组; 对交流分析, 所需求解的是线性方程组; 对瞬态分析, 所需求解的是非线性常微分方程组。因此, 松弛方法也分为相应的三种: 线性方程组的松弛方法、非线性方程组的松弛方法和微分方程组的松弛方法。直接用于求解微分方程组的松弛方法又称为波形松弛法(waveform relaxation)。为了便于将松弛方法与一般标准电路求解方法作比较, 我们简要地回顾一下标准电路方程的求解方法。

在时域分析中, 电路方程是一个常微分方程组, 可表示为如下形式:

$$x' = f(x, t) \tag{9.3.1}$$

或写成隐式形式:

$$F(x', x, t) = 0$$

时域分析算法是在离散的时间点(t_{n+1})上, 用稳定性好的数值积分方法将式(9.3.1)离散化, 把微分方程转化为差分方程, 即一个非线性代数方程组, 我们简写为

$$g(x_{n+1}) = 0 \tag{9.3.2}$$

然后, 用牛顿-拉夫森方法处理方程(9.3.2), 在每个迭代点上, 将方程(9.3.2)转化为线性代数方程来求解, 即

$$T^k x^{k+1} = B^k \tag{9.3.3}$$

最后, 用高斯消去法或 LU 分解法求解方程(9.3.3)。

松弛方法也有三种, 它们分别处理线性方程(9.3.3)、非线性方程(9.3.2)和微分方程

(9.3.1)。松弛方法对电路的形式有较高的要求,一般希望电路能满足如下假定条件:

(1) 所有电阻性元件(包括有源器件),其支路特性方程都以电压为控制变量,电流为受控变量,而且是电压的连续函数。

(2) 储能元件仅限于是两端的压控型电容(可以是非线性的)。

(3) 所有独立电压源都必须有一个节点接地,或可以转换为独立电流源。

(4) 电路中每个节点都通过一个两端电容接地。这个假定对于MOS集成电路通常是可以满足的,因为电路节点与地之间和MOS晶体管端点对地之间总存在有分布电容。

9.3.1 线性松弛方法

线性松弛方法是用松弛法求解线性方程组(9.3.3),即求解 $TX = B$ 的一种迭代方法。它将方程系数矩阵T分裂为三个矩阵之和:

$$T = L + D + U \tag{9.3.4}$$

其中,L为 $n \times n$ 阶严格的下三角矩阵,D为 $n \times n$ 阶的对角矩阵,U为 $n \times n$ 阶严格的上三角矩阵。

线性松弛法有如下两种:

1. 高斯-雅可比方法

高斯-雅可比(Gauss-Jucobi)方法(简称G-J方法)将方程 $(L + D + U)X = B$ 写为如下形式:

$$DX^{k+1} = -(L + U)X^k + B \tag{9.3.5}$$

或

$$\begin{aligned} X^{k+1} &= -D^{-1}[(L + U)X^k - B] \\ &= M_{GJ}X^k + D^{-1}B \end{aligned} \tag{9.3.6}$$

其中k为迭代次数,$M_{GJ} = -D^{-1}(L + U)$

方程组(9.3.5)的第i个方程是

$$a_{i1}x_1^k + \cdots + a_{i,i-1}x_{i-1}^k + a_{ii}x_i^{k+1} + a_{i,i+1}x_{i+1}^k + \cdots + a_{in}x_n^k = b_i \tag{9.3.7}$$

由(9.3.7)式可以看出,方程组中每一个方程只有对角元对应变量为未知变量,如第i个方程中的 x_i^{k+1},其它变量均为已知量。因此,G-J方法的迭代过程是:在第 $k+1$ 次迭代时,由(9.3.5)的第一个方程求出 x_1^{k+1},该方程中 $x_2, x_3, \cdots, x_n$ 用第k次迭代值代替;从第二个方程求出 x_2^{k+1};从第i个方程求出 x_i^{k+1}……直到求出 x_n^{k+1},完成第 $k+1$ 次迭代。这n个方程中,每个方程只有一个未知数;n个方程之间没有任何关联,因此n个方程可以同时求解,或是并行处理。

2. 高斯-塞德尔方法

高斯-塞德尔(Gauss-Seidel)方法(简称G-S法)将方程 $(L + D + C)X = B$ 写成如下形式:

$$(L + D)X^{k+1} = -UX^k + B \tag{9.3.8}$$

或

$$\begin{aligned} X^{k+1} &= -(L + D)^{-1}(UX^k - B) \\ &= M_{GS}X^k + (L + D)^{-1}B \end{aligned} \tag{9.3.9}$$

方程组(9.3.8)中第 i 个方程是

$$a_{i1}x_1^{k+1} + \cdots + a_{i,i-1}x_{i-1}^{k+1} + a_{ii}x_i^{k+1} + a_{i,i+1}x_{i+1}^{k} + \cdots + a_{in}x_n^{k} = b_i \quad (9.3.10)$$

这个方程中仍然只有 x_i^{k+1} 为未知量,已知量 $x_1^{k+1}, x_2^{k+1}, \cdots, x_{i-1}^{k+1}$ 由前(i-1)个方程求得,已知量 $x_{i+1}^{k}, \cdots, x_n^{k}$ 由第 k 次迭代解获得。因此,G-S 方法的迭代过程为:在第 k+1 次迭代时,从方程组(9.3.8)的第一个方程中求解 x_1^{k+1},其余变量 $x_2^{k}, x_3^{k}, \cdots, x_n^{k}$ 由第 k 次迭代获得值代替;从第二个方程中求解 x_2^{k+1},第二个方程中 x_1^{k+1} 由第一个方程新获得解代入,其余时 $x_3^{k}, \cdots, x_n^{k}$ 仍由第 k 次迭代解求得……直到解出 x_n^{k+1},完成第 k+1 次迭代。这 n 个方程中,每个方程也只有一个未知数,但这 n 个方程是互相关联的有序的,每个方程都要利用前面已求解的方程的结果信息,因此这 n 个方程不能同时并行处理。

松弛算法是迭代方法,其收敛条件和收敛速度如下:

(1) 如果 T 是严格的对角占优势的矩阵(或 D 是严格的对角阵),则 G-J 和 G-S 方法收敛于 TX=B 的解。

(2) 若 G-J 和 G-S 迭代收敛,则收敛条件与初值无关,而且至少是线性收敛速度。

(3) 两种方法的运算量是 O(N)(N 为方程阶数)。

因为不是所有电路的电路方程都能保证其系数矩阵是具有对角优势的,故松弛方法不能确保收敛。这也就是为什么在一般电路模拟技术中通常使用高斯消去法或 LU 分解法,而不采用松弛方法求解线性方程的原因。

9.3.2 非线性松弛方法

在非线性方程 $g(x_{n+1})=0$ 阶段采用松弛方法,称为非线性松弛方法。它包括非线性 Gauss-Jacobi(非线性 G-J)方法和非线性 Gauss-Seidel(非线性 G-S)方法。

1. 非线性 G-J 方法

非线性 G-J 方法的算法公式为

$$g_i(x_1^{k}, x_2^{k}, \cdots, x_{i-1}^{k}, x_i^{k+1}, x_{i+1}^{k}, \cdots, x_n^{k}) = 0 \qquad i = 1, 2, \cdots, n \quad (9.3.11)$$

由此求解出 x_i^{k+1}。

2. 非线性 G-S 方法

非线性 G-S 方法的算法公式为

$$g_i(x_1^{k+1}, x_2^{k+1}, \cdots, x_{i-1}^{k+1}, x_i^{k+1}, x_{i+1}^{k}, \cdots, x_n^{k}) = 0 \qquad i = 1, 2, \cdots, n \quad (9.3.12)$$

由此求解出 x_i^{k+1}。

两种方法的收敛判据是 $\| x^{k+1} - x^{k} \| \leq \varepsilon$

这两种方法都是把含有 n 个方程的非线性方程组的求解问题,转化为一系列的一元非线性方程(9.3.11)、(9.3.12)的求解问题,即将 n 个相互紧耦合的非线性方程松弛了。

由于方程(9.3.11)和(9.3.12)是非线性方程,仍需要借用 Newton-Rophson 方法求解,故我们又称这两种方法为 Gauss-Jacobi-Newton(G-J-N)方法和 Gauss-Seidel-Newton(G-S-N)方法。由于(9.3.11)和(9.3.12)是一元非线性方程,采用 N-R 方法时,N-R 迭代过程均是求解一元一次方程,而且 G-J-N 和 G-S-N 两种方法都需有双重迭代循环,外循环是松弛迭代,内循环是 N-R 迭代。一般只需一次 N-R 迭代就可以满足非线性松弛方法的收敛要求。

非线性松弛法与 N-R 迭代法相比较,其优点是: ① 每次迭代只需求解一组方程,而且 G-J-N 法可以采用并行处理同时求解 n 个方程,而 N-R 方法需求解联立方程组。② 便于利用电路的潜伏特性。因此,松弛方法在计算速度和内存要求等方面优于传统迭代算法,适用于求解大规模型数字电路。非线性松弛方法的缺点是: ① 收敛条件较为苛刻。它要求初值充分接近于解,而且要求系数矩阵的特征值位于单位圆内。② 收敛速度是线性的。

9.3.3 波形松弛方法

波形松弛法是在求解微分方程阶段采用松弛方法,是一种时域的松弛算法。它是在一个时间区间上求解电路微分方程,涉及电路变量的时间函数或者说是电容变量的波形。波形松弛法也分为 G-J 波形松弛法和 G-S 波形松弛法。

电路微分方程的一般形式为

$$x' = f(x, t)$$

由于方程中的微分项主要是由电路中电容产生的,而且松弛法要求电路每个节点对地都有电容,因此电路方程要改写为如下形式:

$$C(V)V' + f(V, U) = 0 \tag{9.3.13}$$

其中 V 为节点电位向量,U 为激励源向量,C 是节点电容矩阵。

对于 G-S 波形松弛法,方程(9.3.13)可写为

$$\sum_{j=1}^{i} C_{ij}(V_1^{k+1}, \cdots, V_i^{k+1}, V_{i+1}^{k}, \cdots, V_n^{k})V_j^{k+1} + \sum_{j=i+1}^{n} C_{ij}(V_1^{k+1}, \cdots, V_i^{k+1}, V_{i+1}^{k}, \cdots, V_n^{k})V_j^{k}$$
$$+ f_i(V_1^{k+1}, \cdots, V_i^{k+1}, V_{i+1}^{k}, \cdots, V_n^{k}, U) = 0 \qquad i = 1, 2, \cdots, n \tag{9.3.14}$$

在一次 G-S 波形松弛迭代过程中,逐次求解方程组中的每个方程。在求解第 i 个方程时,第 1 到第(i-1)个变量用最新求得的解波形代替; 第(i+1)个到最后一个变量,用上次迭代求得的解波形代替; 要求解的未知变量是第 i 个变量。

现以 G-S 波形松弛法求解一个简单的 MOS 晶体管电路为例,介绍波形松弛法的应用。

图 9.3.1 所示电路由三个子电路 s_1, s_2 和 s_3 组成,电路中所有电容均为线性电容。该电路的微分方程为

$$(C_1 + C_2 + C_3)V_1' - i_1(V_1) + i_2(V_1, U_1)$$
$$+ i_3(V_1, U_2, V_2) - C_1U_1' - C_2U_2' = 0 \tag{9.3.15a}$$
$$(C_4 + C_5 + C_6)V_2' - C_6V_3' - i_3(V_1, U_2, V_2) - C_4U_2' = 0 \tag{9.3.15b}$$
$$(C_6 + C_7)V_3' - C_6V_2' - i_4(V_3) + i_5(V_3, V_2) = 0 \tag{9.3.15c}$$

使用 G-S 波形松弛法求解方程(9.3.15),其第 k+1 次迭代方程为

$$(C_1 + C_2 + C_3)V_1^{k+1} - i_1(V_1^{k+1}) + i_2(V_1^{k+1}, U_1)$$
$$+ i_3(V_1^{k+1}, U_2, V_2^{k}) - C_1U_1' - C_3U_2' = 0 \tag{9.3.16a}$$
$$(C_4 + C_5 + C_6)V_2^{k+1} - C_6V_3^{k} - i_3(V_1^{k+1}, U_2, V_2^{k+1}) - C_4U_2' = 0 \tag{9.3.16b}$$
$$(C_6 + C_7)V_3^{k+1} - C_6V_2^{k+1} - i_4(V_3^{k+1}) + i_5(V_3^{k+1}, V_2^{k+1}) = 0 \tag{9.3.16c}$$

图 9.3.1　采用 G-S 波形松弛法的电路实例

按照此方程,电路可以改画为图 9.3.2 所示的三个子电路,方程(9.3.16a)、(9.3.16b)和(9.3.16c)分别是图 9.3.2 中(a)、(b)、(c)三个子电路的电路方程。

图 9.3.2　G-S 波形松弛法将电路分为三个子电路

可见,求解(9.3.16)的三个方程也就是分别对图 9.3.2 中的三个子电路进行瞬态分析。即在整个时间区间[0,T],将电路按顺序分解为 s_1,s_2 和 s_3 三个动态子电路。当处理子电路 s_1 时,要考虑 s_2 的近似负载效应 V_2^{k+1},处理 s_2 时,要考虑 s_1 的激励作用 V_1^{k+1} 和 s_3 的负载效应 V_3^{k+1} 等等。也就是说,在处理每个子电路时,都要考虑电路其它部分与该子电路的相互作用,其它子电路的迭代值用最新计算出来的迭代值来近似。这样,用 G-S 波形松弛法对电路进行瞬态分析时,就把一个大规模电路的瞬态问题转化为一些子电路的瞬态问题,降低了电路的分析规模、存储容量和计算时间。

美国加利福尼亚大学 Berkeley 分校于 1982 年发表了一个采用 G-S 波形松弛法的电路模拟程序 RELAX。RELAX 程序用 C 语言编制,在 G-S 波形松弛法基础上做了若干算法改进,能对 MOS 数字集成电路进行相当精确的模拟,加快了算法的收敛速度,并减少了内存要求。RELAX 程序的特点如下:

(1) 允许每个被分解的子电路有多个未知变量。而标准的 G-S 算法,每个子电路只有一个未知变量。

(2) 每个子电路都采用标准的电路模拟技术来求解。即对每个子电路采用隐式数值积分公式将其微分方程离散化,转化为差分方程;再用 N-R 方法将非线性差分方程线性化。由于每个子电路包含的节点数和未知变量数不多,故采用 N-R 方法求解时使用满阵技术而不使用稀疏矩阵技术。

(3) 由于每个子电路之间相互独立,在波形松弛算法中,各个子电路使用的步长可由自身决定。由于某些子电路在大部分时间内处于潜伏状态,因而在大部分时间内都可采用较大步长,而使[0,T]中所需总的积分步数减少,从而提高了分析速度。

(4) 将子电路按信号流向排序,则一般只需一次波形松弛迭代就可获得正确的电路响应。

(5) 首次松弛迭代时不考虑扇出的负载效应,即简单地将后级断开。从第二次迭代以后才采用标准的波形松弛技术,考虑负载效应。这样可以减少迭代次数。

(6) 采用部分收敛算法。当电路中未知向量的子向量在前后两次迭代中,其值没有显著变化,满足收敛条件(即部分收敛了),则在下次迭代中,与该子向量有关的雅可比矩阵部分不再重算。

RELAX 程序的突出优点是,既具有高的模拟精度,又有很高的模拟速度。RELAX 程序与标准电路模拟程序 SPICE 相比较,二者精度基本相同,而在计算时间上 RELAX 程序只有 SPICE 的 20%。而且电路规模越大,RELAX 的优势就越显著。

9.4 多级牛顿算法

多级牛顿算法(Multi-level Newton Algorithm)是一种适用于大规模非线性电路时域分析的迭代算法。

对于含有多个子电路的大规模电路,多级牛顿法的实质就是在给定电路输入和激励的条件下,先对每一个子电路进行低一级的牛顿迭代,计算出各子电路的端口电压和电流,然后在此基础上对整个电路进行高一级的牛顿迭代,求出全部电路的电压和电流信息。我们以含有一个子电路的电路为例,介绍多级牛顿算法的原理,它很容易推广到含有多个子电路的情况。

设要分析的电路为 N,N 中含有一个子电路 s,N 中除 s 以外的电路称为剩余电路,电路 s 通过(l+1)个节点与剩余电路相互作用,这(l+1)个节点中有一个是参考节点。令 x 是子电路 s 的内部变量,$x \in R^x$;y 是 s 的输出变量,$y \in R^l$;u 为 s 的激励或输入变量;那么当进行时域分析时,在 t_n 时刻,s 通过(l+1)个节点与剩余电路作用,$s(t_n)$ 可以由一组非线性代数方程来描述:

$$H_n(u_n, x_n, y_n) = 0 \tag{9.4.1}$$

这里 $H \in R^{x+l}$。假定对于每一个有意义的 u,(9.4.1)式都有一个与之对应的解 y,那么(9.4.1)式就隐含地描述了从 u 到 y 的映射关系:

$$y = G(u) \tag{9.4.2}$$

这就是子电路 s 的宏模型公式。那么,在 t_n 时刻,整个电路的特性方程可由如下非线性方程表示:

$$F(u, G(u), w) = 0 \tag{9.4.3}$$

其中，w 是电路中不与 s 相互作用的电路变量，$w \in R^p$，则 $F \in R^{l+p}$。

用牛顿法求解(9.4.3)式，得到如下牛顿迭代方程：

$$dF(u^k, G(u^k), w^k) \begin{bmatrix} \Delta u \\ \Delta w \end{bmatrix} + F(u^k, G(u^k), w^k) = 0 \tag{9.4.4}$$

其中，dF 为 F 在 t_n 时刻的雅可比矩阵，$\Delta u = u^{k+1} - u^k$，$\Delta w = w^{k+1} - w^k$。(9.4.4)式就是电路级牛顿迭代方程，也称为高一级牛顿迭代。按照(9.4.4)式进行迭代计算时，必须首先计算出 $G(u^k)$ 和 $dG(u^k)$。因此，应该对(9.4.1)式采用牛顿迭代算法，即

$$dH(u^k, x^{k,j} y^{k,j}) \begin{bmatrix} \Delta x \\ \Delta y \end{bmatrix} + H(u^k, x^{k,j}, y^{k,j}) = 0 \tag{9.4.5}$$

其中，dH 是 H 在 t_n 时刻的雅可比矩阵，$\Delta x = x^{k,j+1} - x^{k,j}$，$\Delta y = y^{k,j+1} - y^{k,j}$。(9.4.5)式是子电路一级的牛顿迭代方程，也称为低一级牛顿迭代方程。

由此可见，多级牛顿算步骤可归纳如下：

(1) 在给定子电路输入变量 u 初值的情况下，构造 $H(u, x, y)$ 和 $dH(u, x, y)$，形成子电路的牛顿迭代方程(9.4.5)，并求解出 Δx 和 Δy。这是子电路一级的牛顿迭代过程，它可以推广到多个子电路的情况，即 $H_i(u_i, x_i, y_i)$，$i = 1, 2, \cdots, m$。

(2) 由(9.4.2)式计算出 $G(u) = y$。

(3) 建立整个电路的牛顿迭代方程，即构造 $dG(u)$，$F(u, G(u), w)$ 和 $dF(u, G(u), w)$，形成(9.4.4)式，并求解出 Δu 和 Δw。这是电路级(高级)牛顿迭代过程。

采用多级牛顿算法的优点如下：

(1) 该算法保持了经典牛顿算法的局部二次收敛性。可以证明，只要子电路牛顿迭代收敛，由此所求得的子电路宏模型 $G(u)$ 和 $dG(u)$ 就是精确的。将其代入整个电路牛顿迭代式，满足电路级收敛条件时，整个电路的牛顿迭代就具有局部二次收敛性。

(2) 多级牛顿算法是先分别对每一个子电路进行牛顿迭代，其节点数要远远小于整个电路总节点数。设每个子电路的节点数都相同，且等于 p，那么求解每个子电路的运算量约为 $\frac{1}{3}p^3$，k 个子电路总计算量为 $\frac{1}{3}kp^3$。求解整个电路的牛顿迭代方程(9.4.4)的维数等于整个电路的独立节点数 n 减去每个子电路的节点数，即 $(n - kp)$，其运算量约为 $\frac{1}{3}(n - kp)^3$。当电路中包含子电路较多时，方程(9.4.4)的维数会比电路总节点数小得多。例如，$n = 200$，$p = 20$，$k = 8$ 时，k 个子电路的求解计算量 $\frac{1}{3}kp^3 = 2.13 \times 10^4$，方程(9.4.4)的维数 $(n - kp) = 40$，计算量 $\frac{1}{3}(n - kp)^3 = 2.13 \times 10^4$，故总的计算量为 4.26×10^4。如果按通常牛顿法求解含 200 个节点的电路方程，其总的运算量约为 $\frac{1}{3}n^3 = 2.67 \times 10^6$。可见，当电路中含有较多子电路时，多级牛顿法运算量明显减小。

(3) 多级牛顿算法特别适于采用并行处理。对于每一个子电路的低一级牛顿算法，可以同时并行处理，当所有子电路都收敛以后，再总的进行高一级牛顿迭代。

9.5 混合仿真技术

混合仿真技术是随着当前大规模电子系统仿真的需求而兴起的一门新的仿真方法。统计资料表明,在无线电通信、多媒体、人工智能和消费类电子设备等领域,每年生产出的专用集成电路和系统中各种不同类型的混合集成产品约占总的集成电路产品的 30% 左右,而且混合集成产品的种类和需求量逐年增加,估计到 2000 年这个比例能达到 60%。混合集成产品的增长必然对混合仿真工具提出更高的要求,并促进混合仿真算法和工具的迅速发展。混合仿真技术已经成为当前 EDA 领域十分热门和活跃的一个研究课题。在这一节中,我们除了介绍各种不同类型的混合仿真方法外,将着重介绍目前集成电路中应用最广泛的数字电路和模拟电路的混合仿真方法。

混合集成电路产品和混合仿真工具所涉及的面相当广泛,我们感兴趣的仅仅是与电子产品有关的混合技术。在电子设计自动化领域中,混合仿真技术包括的内容如下所述。

1. 不同学科的混合仿真(Mixed-Discipline)

(1) 电路与机械的混合仿真

机电一体设计是当前十分时髦的一个研究领域。几乎多数的机电系统都是由电子设备控制的,例如:录音机、录相机、CD 机以及各种其它类型电机的控制、伺服系统,冰箱、洗衣机、空调等家用设备的控制系统,汽车电子设备等等。在设计这种控制集成电路时,要采用机电一体化的设计方法。在电路设计时要紧密地结合机械设备的特性,除了保证控制功能外,还要考虑控制的可靠性和效率,如过载保护、抗干扰能力等等。图 9.5.1 中绘出了一个简单的机电混合设计的示意图。

图 9.5.1 机电混合设计示意图

机电混合仿真的技术关键是:① 如何描述和构造机械部件的数学模型。② 如何实现机电之间的数据转换,即描述和构造机电之间的接口模型。

(2) 电路与光学设备、器件的混合仿真

光和电历来是紧密相关的。随着通信事业的飞速发展,光纤、光缆技术在通信领域的广泛应用,人们迫切需求能实现光电一体设计的混合仿真工具。它也是通过建立光学系统或器件的数学方程和光电之间的数据转换来实现的。

(3) 电系统的热分布仿真

集成系统和电路的功耗是集成电路可靠性的一个重要的标志,因此计算电路的功耗、热分布特性也是 EDA 的一个重要内容。目前不少 EDA 软件中都有集成电路芯片和印刷电路板热分布分析的功能,它通过电路的功率分析和有限元分析方法来实现,属于电学和传热学的混合仿真技术。

(4) 电与磁的混合仿真

磁性材料器件本来就是电器件的一部分,不少电路仿真工具中都含有磁性材料和器件的模型,例如不同材料、不同规格、形状、尺寸的磁芯和变压器等等。另外,随着集成电路工作频率的升高和功率的增大,电磁之间的辐射干扰是电路设计中面临的一个很大问题,因此如何模拟集成电路芯片内部和 PCB 板上的磁场分布特性是当前 EDA 领域的一个热门课题。

美国 Analogy 公司的 EDA 软件 Saber 是一个在混合仿真上能力较强的仿真工具。它除了能对模拟和数字电路分别进行仿真以外,还可以进行机电一体设计,电源、磁性材料设计,数字/模拟混合集成设计,化学电源设计及汽车电子器件设计等等,是一个实现不同学科技术、不同模式、不同层次的混合仿真工具。

2. 不同分析域的混合仿真(Mixed-Domain)

所谓 Mixed-Domain 仿真是指电路在时域、频域和离散时域(或称数据采样域)的混合仿真。

时域和频域的混合仿真主要是针对电路中含有高频和微波器件的情况。微波器件一般用频域 S 参数表示,当需要进行时域分析时,就需要将网络和器件的 S 参数转换为电路参数,因此需要时域和频域的混合仿真。

连续时域和离散时域的混合仿真是针对一些含有数字信号处理电路和开关电容电路的分析与设计。这些电路在时钟作用周期内,仅在时钟定义的时间点上求解电路特性。通常有专门用于数字信号处理和开关电容电路分析的软件工具,将这类分析工具与电路分析工具结合在一起,构成连续和离散的时域混合仿真工具。

3. 不同层次的混合仿真(Multi-level)

不同层次的混合仿真最初是由逻辑仿真的混合级模拟发展起来的。在逻辑仿真中,按照被模拟器件抽象的层次,一般分成四级:开关级、门级、功能级和寄存器传输级。近年来,由于逻辑综合工具的出现,又增加了一个最高级:行为级。

开关级模拟,是以晶体管为基本元件,每个晶体管被看成一个独立的、理想的开关。开关级模拟可以模拟双向传输门、不同的信号强度等问题。

门级模拟,是以逻辑门:与门、或门、异或门等等为基本元件(也允许是由这些基本门组成的"宏门"),将逻辑门行为和内连关系描述出来。

功能级模拟,是以功能块:计数器、寄存器、锁存器和触发器等为基本单元。

寄存器传输级模拟,是以存储器、组合逻辑电路和总线部件等为基本单元,以计算机系统为模拟对象,检查计算机指令系统的流程,即数据在有关各寄存器中的传输情况。

行为级模拟,是最高抽象层次的模拟,基本元件可以是任意一个子模块电路实体。它以硬件描述语言(VHDL)为输入形式,每个实体可以是结构描述,也可以是行为式描述。

这样,一个电路的模型就是一系列行为的集合,并且用进程(process)来表示行为,将电路的工作过程表现为若干并行的进程。因此,就可以在系统级(行为级)上对电路进行仿真,并能完成从行为到电路结构的映射和优化,这就是电路综合。

多层次逻辑仿真是将上述几种不同层次的模拟联合在一起形成混合逻辑仿真器,所能模拟的器件可以从行为级、寄存器传输级直到门级或开关级。用混合仿真器进行电路分析和设计时,根据不同的功能和精度需求,对电路中某些模块可以做高一级的行为级或寄存器级模拟,对某些有特殊精度要求的模块做门级甚至是开关级的模拟。目前,几乎所有EDA公司提供的逻辑仿真工具都具有多层次混合模拟的功能。当然,真正具有逻辑综合能力,能以VHDL语言描述行为级模型的逻辑仿真工具,还数Synopsys, Mentor Graphics, Cadence和Viewlogic等几个国际上享有盛誉的公司。

4. 不同模式的混合仿真(Mixed-Mode)

在集成系统和集成电路的仿真中,不同模式的混合仿真是指:数字电路和模拟电路的混合,数字电路、模拟电路和数字信号处理(DSP)的混合,电路和器件的混合等等。这是目前混合仿真领域中应用最多的一种混合方式。

(1) 数字电路、模拟电路和数字信号处理的混合仿真

由于数字电路具有形式和结构简单、规整,易于抽象和集成,并有功耗小、抗干扰性能强等优点,故以数字计算机和数字信号处理为代表的数字VLSI近年来发展迅速。人们对数字系统的需求和拥有量越来越大,应用面越来越广,对数字信号的处理能力也远远超过模拟信号。然而,我们生活的自然界是一个模拟世界,人与人、人与自然界之间的信息交换和处理就不可避免地需要数字信息和模拟信息之间频繁、有效地转换。因此,人们对于数/模/DSP混合系统的需求也就越来越迫切。图9.5.2示意地给出了一个数/模/DSP混合系统的实例。这是一个有信号压缩机制的CD系统,该系统处理的信号是模拟信号,最终提供给收听者的信号也是模拟信号,而中间的处理信号是数字信号。为了便于传输和存储,还采用了压缩编解码的信号处理电路。目前这样一个系统完全可以在一个IC芯片上实现。类似的数/模/DSP系统还有很多,例如传感器、换能器、机器人、多媒体系统、各种人工智能系统、声象处理系统、通信网络等等。

图9.5.2　数模混合电子系统

由此,对数/模/DSP仿真工具也提出了更高的要求,图9.5.3表示了这样一个混合

仿真的示意图。设计输入是图形输入和 VHDL 语言输入,先进行系统级和行为级仿真,并对芯片做功能上的划分,分为数字电路、模拟电路和 DSP 三部分,分别进行仿真验证,最后进行芯片版图的布局布线。

图 9.5.3　数/模/DSP 混合集成设计示意图

(2) 数字电路和模拟电路的混合仿真

数/模混合仿真是前面所述的数/模/DSP 仿真的一个子集,但就仿真工具而言,数/模混合仿真工具在方法和实现上更加成熟,应用也极为普遍。

数/模混合仿真是指,一个集成电路中同时含有数字电路和模拟电路时,对它们的整体特性进行仿真。数/模混合仿真的关键问题是模拟和数字之间的数据转换。模拟电路的输入输出结果大多是波形,近似为连续变量;而数字电路的输入输出是 0, 1, x 电平。因此,连续波形与电平之间的数据转换,需要设计专门的 A/D 和 D/A 接口元件模型来实现。另外,有效地解决数/模混合仿真的精度和速度也是十分重要的。

目前常用的数/模混合仿真方法有以下三种:

1) 以模拟电路为核心的混合方法

这种以模拟电路仿真为核心的数/模混合方法是在模拟电路仿真程序中加入数字电路的模型,例如逻辑门、寄存器,甚至是行为模型。算法以模拟电路为主,即连续的波形分析。模拟电路和数字电路之间有 A/D, D/A 接口元件。这种混合方式一般用于主要是模拟电路,而只含有少部分数字电路的情况,它能够保证模拟电路的计算精度,但对数字电路(尤其是较为复杂的数字电路)仿真不是太有效。MicroSim 公司的高版本 PSpice 程序(版本 5.0 以上)就是这样一个混合软件包,它在原来 PSpice 的基础上增加了一系列逻辑门和寄存器模型以及 A/D, D/A 接口元件,可以进行一般的数/模混合仿真。

2) 以数字电路为核心的混合方法

这种以数字电路为核心的混合仿真方法是在数字电路仿真程序上扩展模拟电路模型。算法主要是逻辑模拟所采用的事件驱动和选择性跟踪方法。对电路的某一小部分可以进行细致的波形分析,而主要部分是逻辑仿真。例如美国加州大学 Berkeley 分校的

SPLICE 程序,就是一个用于 MOS 大规模集成电路的混合仿真程序,它能有效地进行数字/模拟电路的时域分析,在模拟电路部分分析精度上低于模拟仿真工具,但分析速度较高。图 9.5.4 是一个简单数/模混合电路图,图 9.5.5 中表示了图 9.5.4 电路所需 A/D 接口元件。

图 9.5.4　数/模混合电路实例

图 9.5.5　数模混合仿真中的 A/D 接口元件

3) 耦合方式的数/模混合仿真

耦合方式(Glued Approach)数/模混合仿真的特点是:① 在混合仿真工具中数字电路仿真程序和模拟仿真程序是相互独立存在,而且独立工作的。② 监控和进程调度程序的作用是将数/模两部分程序进行时间上的同步和数据的转换。

图 9.4.6 是一个典型的耦合方式数/模混合仿真的框图。

这种混合仿真方法是目前大多数 EDA 公司的商用数/模混合仿真工具所采用的方法。它的优点是可以充分发挥和利用数字仿真和模拟仿真各自的优势,保证数字电路和模拟电路仿真的精度和速度,从而提高整个混合仿真的准确性和效率。这种混合仿真的关键是数字电路与模拟电路在事件和时间上的同步技术,以及进程调度管理方法。

(3) 电路与器件的混合仿真

随着集成电路集成度的增加和新器件(如砷化镓器件、光电器件等)的不断涌现,人们

图 9.5.6　耦合式混合仿真框图

迫切需要强有力的仿真技术和工具对新器件和由新器件组成的电路进行设计与验证。以往在研究新器件和应用新器件设计新电路的过程中,若想预估或评价由新器件组成的电路特性和响应是十分困难的,因为应用电路仿真工具进行电路模拟时,必须先具备器件模型和模型参数,而建立器件模型和提取模型参数都是十分复杂、繁琐的工作,并且在新器件制造出来以后才能进行。一般,从新器件设计开发到建立器件的数学模型常常需要几年时间,这就形成了当前新器件与新电路发展应用的一个“瓶颈”。80 年代末到 90 年代初,美国首先提出了电路与器件的混合仿真方法,直接用器件仿真软件作为电路仿真工具中的器件模型,绕过了建立器件模型和模型参数提取的复杂过程。这样,在器件设计阶段,就可以利用混合仿真软件同时进行器件模拟和电路模拟,从而加速了新器件和新电路的设计周期。

图 9.5.7 是用电路仿真工具 SPICE 和一个二维器件仿真软件 PISCES(美国 Stanford 大学开发)组合形成的一个电路/器件混合仿真工具的框图。在电路仿真工具 SPICE 中增加了一个数值器件模型,它将器件仿真软件 PISCES 计算出的端电流、电导、

图 9.5.7　电路/器件混合仿真框图

电容等信息转化为电路模拟可以接受的数值器件参数，并写到电路方程中去。器件仿真软件 PISCES 用 SPICE 给出的电路中器件的端电压、端电流信息作为初始条件，通过给定的器件结构、材料、掺杂浓度等条件求解半导体方程，计算出器件的端电流、电导等，再送回给 SPICE。

第 10 章　电路的优化设计方法

10.1　电路优化设计概述

前面介绍的直流非线性分析、交流小信号分析、瞬态分析、灵敏度分析和容差分析等等都属于电路计算机辅助分析(CAA)方面的内容,它们是在给定电路的拓扑结构和电路中元器件参数的前提下,分析电路的状态和各种特性。如果计算机计算出的分析结果不符合电路设计的特性指标要求,则需要靠人工的方法调整、修改电路的元件参数以及电路的结构,并经过多次反复修正,最后使电路的特性达到设计要求。

电路的优化设计方法,应包括自动设计电路的拓扑结构和自动确定电路的元器件参数两方面。一般是将最优化算法和上述的电路 CAA 分析程序相结合,通过进一步编程实现一个电路优化设计工具,我们称之为电路优化设计(CAD)或电子设计自动化(EDA)工具。对于数字电路,可以很容易地抽象出逻辑门、加法器、减法器、寄存器等不同层次的逻辑单元,这种抽象极大地促进了数字电路的设计自动化,从高层次的自动综合,到最低层次的集成电路版图的布局布线,都有成熟、实用的自动设计软件工具。而模拟电路的情况则要复杂得多,由于模拟电路种类繁多,电路结构千差万别,要实现电路结构的自动设计十分困难。模拟电路的自动设计技术和工具也远没有数字电路的相应技术成熟。尽管有不少科技工作者在模拟电路自动设计方面进行研究和探索,对一些简单电路实现了自动综合,模拟电路的硬件描述语言 AHDL 也和 VHDL 一样成为设计标准,但与真正的自动综合设计还有较大差距。目前实用的模拟电路优化设计工具所能实现的,还是在给定电路拓扑结构和电路元件初值的情况下,按照电路性能指标要求,优化设计电路元器件参数的最佳值。

利用 CAD 技术进行电路优化设计的过程可用图 10.1.1 说明,它包括以下几方面内容和主要步骤:

1. 选择电路的拓扑结构和给出电路元件的初始值。根据电路设计者的经验选择合理的电路拓扑结构,以及较好的元件初始猜测值。元件初始值选择得越接近最佳值,则优化越容易,越快。对于较为复杂的电路,最好先进行电路分析,帮助设计者选择适当的电路结构和元件初始值。

2. 根据电路的优化目的,确定优化参数、约束条件和目标函数。

3. 利用电路 CAD 软件计算目标函数。因为对于许多复杂的电路特性要求,很难写出目标函数的显式表达式,因而要用电路 CAD 程序分析出电路的特性,比如电路的增益、带宽、输入输出阻抗、等效噪声系数、时域波形等电路特性。电路分析可以采用前面各章所述的方法: 直流非线性分析、频域分析、瞬态分析、容差分析等等。

4. 利用优化算法,判断目标函数是否已足够小到满足误差要求,若不满足,则根据优化算法确定修改元器件参数的策略。例如,可根据目标函数对元器件参数的梯度,以目标

图 10.1.1　电路优化设计框图

函数下降的方向作为元件参数的修改方向,对元件参数进行调整。然后再进行电路分析和误差比较,经过多次迭代,直到目标函数满足要求为止。

由此可见,电路优化设计是将最优化方法与电路 CAD 分析程序结合起来而实现的。

最优化设计方法本身属于纯数学的计算方法。最优化设计方法的数学描述是

$$\begin{aligned} &\min\ F(P) \\ &g_i(P) \geq 0 \qquad i = 1, 2, \cdots, l \\ &h_j(P) = 0 \qquad j = 1, 2, \cdots, k \end{aligned} \tag{10.1.1}$$

式中 $F(P)$ 是目标函数,用以评价设计好坏的标准,$P = (p_1, p_2, \cdots, p_m)^T$ 是元件参数向量。$g_i(P) \geq 0$ 和 $h_j(P) = 0$ 是约束条件,它们分别是不等式约束和等式约束,通过它们对目标函数进行某些限制。例如,在电路设计中元件参数值大多是非负的,另外还需服从电路基本定律 KCL 和 KVL 等等。这些约束条件可以是显式的,也可能是隐式的。因此,电路最优化设计就是在一定约束条件下求目标函数的极值问题。

最优化方法所涉及的面很广,可以有各种不同的要求和考虑,例如:

(1) 有没有约束? 有约束的话,是等式约束还是不等式约束?

(2) 是确定性的还是随机性的最优问题?

确定性最优问题中,每个变量取值是确定的,可知的。而随机(或概率)最优问题中,某些变量的取值不确定,服从于一定的统计概率分布。

(3) 目标函数是线性的还是非线性的?

如果目标函数和所有约束条件都是线性的,则称为线性最优化,或线性规划。如果目标函数或约束式是变量的非线性函数,则称为非线性最优化,或非线性规划。

(4) 是单目标优化还是多目标优化?

如果是多目标优化问题,各个目标要求常常相互矛盾,不能求得绝对最优解,而只能获得有效解。

最优化方法的具体分类如下:

(1) 解析法。解析法也称为间接法。一般采用求导数的方法计算函数的极值,它只适用于简单函数和具有明确数学表达式的函数。

(2) 直接法。当目标函数较复杂或不能用显函数表示时,我们要用直接搜索的方法经过若干次迭代搜索到最优点,它分为一维搜索和多维搜索问题。

1) 一维搜索: 主要有斐波那西法、黄金分割法和多项式插值法等等。

2) 多维搜索: 主要有坐标轮换法、步长加速法、方向加速法和单纯型法等等。

(3) 梯度法。利用目标函数的梯度来判断优化方向,也是一种直接法。

1) 无约束梯度法: 主要有最速下降法、拟牛顿法、共轭梯度法和变尺度法。

2) 有约束梯度法: 主要有可行方向法和梯度投影法等等。

3) 化有约束问题为无约束问题: 主要有罚函数法和直接法。

(4) 统计优化方法。这是一种随机优化方法。它以蒙特卡罗为基础,求解合格率的最优问题。

实际应用中,常常是几种优化方法配合使用。例如同时使用一维搜索、无约束梯度法以及罚函数法等。而且所求得的最优解,从全局来看不一定是最优,而从邻近的小范围看是最优(称为局部最优解)。

在本章中,我们只介绍在电路优化设计中常用的几种优化方法。另外,近年以模拟退火算法为代表的全局优化算法在电路优化中受到相当的关注,我们也将做简单的介绍。

10.2 目 标 函 数

目标函数是由电路特性的误差函数组成的,它是电路实际特性与设计要求特性之间误差的量度,是评价电路设计好坏的定量指标。电路最优化的实质是使误差函数逐步降为极小的过程,当实际电路特性与理想特性的误差函数已达到可能的最小值时,则电路达到最优。因此,优化设计就是求目标函数的极小值。如何由实际电路特性和给定的理想特性构成目标函数,是电路优化的一个重要环节。

1. 目标函数的表达式

目标函数的构造与描述比较复杂,因电路和性能指标的要求而异,没有一定的规律可循。因此,不可能给出目标函数的统一表示形式,只能针对具体不同的电路设计问题,给出不同的描述方式。

(1) 对电路的频响特性进行优化设计,其幅频特性如图 10.2.1 所示,图中矩形曲线表示设计要求的理想特性,峰形曲线表示实际响应特性。这种以频率响应作为设计要求的目标函数可以用各频率点的误差函数来构造,形式如下:

$$F(P,\omega)=\sum_{i=1}^{m}W(\omega_i)\left[T(P,\omega_i)-T(P,\omega_i)\right]^{k}\Big]^{1/k} \qquad (10.2.1)$$

其中,$F(P,\omega)$ —— 目标函数;

图 10.2.1　误差函数示意图

P ——待优化的电路元器件参数向量, $P=[p_1, p_2, \cdots, p_n]^T$, 表明有 n 个元件参量;

ω——频率, 通常是离散的频率采样值;

T(P, ω) ——电路的频率响应特性;

T(P, ω) ——设计要求的理想频率响应特性;

W(ω) ——频率的加权函数;

m——频率采样点数;

k——误差函数的指数, $1 \leq k \leq \infty$。

目标函数中几个参数的选取原则如下:

1) 频率采样点数 m

m 可以是频率采样点数, 也可以是时间采样点数, 或者其它变化参量的采样点数。采样点的数量选择因优化对象而异。例如频率特性优化时, 对具有较平坦频响特性的电路, 在全频带选取 5～10 个频率点就够了; 若频响特性较为复杂, 则应选取较多的频率点。

2) 误差函数的指数 k

误差函数的指数 k 值选取范围很宽。当选 k= 1 时, 目标函数是代数和形式的, 即

$$F(P, \omega) = \sum_{i=1}^{m} W(\omega_i) \left| T_i(P, \omega) - T(P, \omega) \right| \qquad (10.2.2)$$

此类目标函数的优化过程平缓, 速度也快。但当函数中出现尖峰, 或平坦增益特性的频带边缘处达不到误差要求时, 我们可以通过在这种超差的采样点处加大权函数 W(ωi)的方法来克服。

误差函数指数的第二种选择是 k→∞。在这种情况下, 误差函数中数值越大的分量影响越大, 目标函数逼近于误差函数中的最大值。这种最大误差形式的目标函数表达式为

$$F(P, \omega) = \max_{i=1\text{-}m} \left[\left| T_i(P, \omega) - T(P, \omega) \right| \right] \qquad (10.2.3)$$

可见 k 指数的加大是对误差函数中数值大的分量自动加权。但当 k 指数较大时, F(P, ω)值可能过大, 而导致计算机溢出, 故通常把目标函数改写为式(10.2.1)的形式, 即

$$F(P, \omega) = \sum_{i=1}^{m} W(\omega_i) \left| T_i(P, \omega) - T(P, \omega) \right|^k \Bigg]^{1/k}$$

误差函数指数最常选用的数值是 k= 2, 此时目标函数为平方和的形式, 即

$$F(P,\omega)=\sum_{i=1}^{m}W(\omega)[T(P,\omega)-T(P,\omega)]^{2}\quad^{1/2} \tag{10.2.4}$$

3) 加权函数 $W(\omega)$

选择加权函数 $W(\omega)$ 的目的是改善设计效果，要根据具体设计指标选定。$W(\omega)$ 是个正实数，根据电路设计指标要求，在不同的采样点可选取不同的数值，用以权衡各采样点对性能的要求。在实际电路的频域设计中，一般难于保证全频域和全部指标都达到最优，因为目标函数所涉及的各指标之间有些是相互矛盾或相互牵制的。例如频带与增益，动态范围与最大输出功率之间都存在牵制。我们可以通过选用适当的加权系数，对各指标权衡取舍，以保证重要指标或重要频域内的特性要求。

(2) 对电路的时域特性进行优化，其目标函数可以表示为实际瞬态响应特性 $V(P,t)$ 和理想时域特性 $V(t)$ 的误差函数的形式，即

$$F(P,t)=\sum_{i=1}^{m}\left\{W(t_i)[V(P,t_i)-V(t_i)]\right\}_i^2 \tag{10.2.5}$$

(3) 对电路的静态工作点进行优化设计，设计参数可以是电路节点电位 V_i，某些感兴趣的支路电流 I_j，以及电源功耗等等。优化目标函数可表示为

$$F(P)=\sum_{i=1}^{n}\frac{W_{V_i}(V_i-V_i)}{\left\|V_i\right\|_i}^{2}+\sum_{j=1}^{m}\frac{W_{I_j}(I_j-I_j)}{\left\|I_j\right\|_i}^{2}+\sum_{k=1}^{l}W_k(I_sU_s)^2 \tag{10.2.6}$$

这里节点电位和支路电流用相对误差函数表示，式(10.2.6)中右边第三项表示要求电源功耗最小，U_s 和 I_s 分别表示电源电压和电流。W_{V_i}，W_{I_j} 和 W_k 为加权系数。这实际上是一个多目标优化问题。

2. 目标函数的极值

最优化方法的目标是寻找目标函数的极小值，这就需要首先分析函数的极值特性，掌握函数极值的规律，以便有效地进行最优化设计。

(1) 一元函数极值

对于一元函数 $F(p)$，我们希望找到一个极小点 $p=p^*$，使得对所有的 p 均有

$$F(p^*)\leq F(p) \tag{10.2.7}$$

这个极小点存在的充分和必要条件是一阶导数等于零及二阶导数大于零，即

$$\left.\frac{F(p)}{p}\right|_{p=p^*}=0 \tag{10.2.8}$$

$$\left.\frac{^2F(p)}{p^2}\right|_{p=p^*}>0 \tag{10.2.9}$$

图 10.2.2 所示是一元函数在定义区间内有一个极值点的情况。实际上函数在定义区间内通常可能有几个极小值，即 $F(p)$ 是多峰函数。如图 10.2.3 所示，在定义域 $[a,b]$ 区间内，有三个极小点 p_1，p_4，p_6，其中 p_6 为全局极小点，p_1 和 p_4 是局部极小或相对极小点。其中 p_2 和 p_5 是相对极大点，而 p_3 点称为驻点或拐点。极值点一定是驻点，但驻点不一定是极值点。在最优化方法中，如果是极大值问题，一般将其转化为极小值问题来求解。

(2) 多元函数极值

图 10.2.2　一元单峰函数　　图 10.2.3　一元多峰函数

我们先推导二元函数的极值条件,并以此推广到多元函数(n 维变量)的极值条件。

将二元函数 $F(P)$ 展成台劳级数,并略去高阶导数项,得

$$F(P+\Delta P)=F(P)+\frac{F(P)}{p_1}\Delta p_1+\frac{F(P)}{p_2}\Delta p_2+\frac{1}{2}\left[\frac{\partial^2F(P)}{p_1^2}\Delta p_1^2+2\frac{\partial^2F(P)}{p_1\,p_2}\Delta p_1\Delta p_2+\frac{\partial^2F(P)}{p_2^2}\Delta p_2^2\right] \tag{10.2.10}$$

写成矩阵形式,则有

$$F(P+\Delta P)=F(P)+\left[\frac{F(P)}{p_1}\quad\frac{F(P)}{p_2}\right]^{\mathrm{T}}\Delta P+\frac{1}{2}\Delta P^{\mathrm{T}}\begin{bmatrix}\dfrac{\partial^2F(P)}{p_1^2} & \dfrac{\partial^2F(P)}{p_1\,p_2}\\ \dfrac{\partial^2F(P)}{p_1\,p_2} & \dfrac{\partial^2F(P)}{p_2^2}\end{bmatrix}\Delta P \tag{10.2.11}$$

式中 $P=[p_1\quad p_2]^{\mathrm{T}}$。

由此可推广到一般的 n 元函数。将 n 元函数 $F(P)$ 的台劳级数展开式写为如下形式:

$$F(P+\Delta P)=F(P)+[\grad F(P)]^{\mathrm{T}}\Delta P+\frac{1}{2}\Delta P^{\mathrm{T}}H\,\Delta P \tag{10.2.12}$$

式中

$$P=[p_1\quad p_2\quad p_3\quad\cdots\quad p_n]^{\mathrm{T}}$$

$$F(P)=\left[\frac{F(P)}{p_1}\quad\frac{F(P)}{p_2}\quad\cdots\quad\frac{F(P)}{p_n}\right]^{\mathrm{T}}$$

$$H=\begin{bmatrix}\dfrac{\partial^2F(P)}{p_1^2} & \dfrac{\partial^2F(P)}{p_1\,p_2} & \cdots & \dfrac{\partial^2F(P)}{p_1\,p_n}\\ \dfrac{\partial^2F(P)}{p_2\,p_1} & \dfrac{\partial^2F(P)}{p_2^2} & \cdots & \dfrac{\partial^2F(P)}{p_2\,p_n}\\ \vdots & \vdots & \vdots & \vdots\\ \dfrac{\partial^2F(P)}{p_n\,p_1} & \dfrac{\partial^2F(P)}{p_n\,p_2} & \cdots & \dfrac{\partial^2F(P)}{p_n^2}\end{bmatrix} \tag{10.2.13}$$

$\grad F(P)$ 称为目标函数 $F(P)$ 在 P 点的梯度,矩阵 H 为 $n\times n$ 维的二阶偏导数矩阵,称为海森(Hessian)矩阵。

如果 $P=P^*$ 是极小点,则

$$\nabla F(P^*) = 0 \tag{10.2.14}$$

此时式(10.2.12)变为

$$F(P^* + \Delta P) - F(P^*) = \frac{1}{2}\Delta P^T H \Delta P \tag{10.2.15}$$

由于 P^* 是极小点,一定有 $F(P^* + \Delta P) > F(P^*)$,故有

$$\Delta P^T H \Delta P > 0 \tag{10.2.16}$$

它等价于二阶偏导数矩阵 H 在向量 ΔP 的全部区域内是正定的。

因此 P^* 为极小点的充分和必要条件是:① 梯度 $\nabla F(P) = 0$;② 二阶偏导数矩阵 H 是正定的。

10.3 单变量函数优化

如果目标函数或约束条件不能用显函数形式表达出来,则需要用数值方法求解最优化问题,电路的优化设计就具有这一特点。数值法的基本思想是经过一系列迭代,产生点的序列$\{P^k\}$,使之逐步接近最优点。

迭代步骤如下:

(1) 从初始猜测点 P^0 开始;

(2) 寻找一个合适的方向 $S^k(k = 0, 1, \cdots)$,S^k 为第 $k+1$ 次迭代的搜索方向;

(3) 沿 S^k 方向向前进一步的步长设为 λ,求合适的步长 λ;

(4) 由 $P^{k+1} = P^k + \lambda S^k$ 得到新的点 P^{k+1},它应当比原来的点 P^k 更接近最优点;

(5) 检验 P^{k+1} 是否最优,若最优则停止迭代;否则 $k = k+1$,转(2)步骤继续迭代。

由上述可知,用数值方法求解最优化问题的关键在于确定搜索方向 S^k 和步长 λ。我们先介绍单变量函数最优化问题,即求最优步长 λ 的方法,称之为线搜索方法,或一维搜索方法。然后,我们再介绍多维搜索,即确定搜索方向 S^k 的方法。因此,一维搜索问题变为求单变量函数 $\varphi(\lambda)$ 的极小值,$\varphi(\lambda)$ 的定义是

$$\varphi(\lambda) = f(P^k + \lambda S^k) \tag{10.3.1}$$

一维搜索的方法很多,归纳起来可分为两类:一类是函数逼近法,也称插值法。它是采用某种较为简单的函数来近似单变量目标函数,通过求近似函数的极值来逼近 $\varphi(\lambda)$ 的极值。典型的函数逼近法有二次插值法和三次插值法。另一类是试探法。它是按某种规则进行试探,通过试探不断缩小 $\varphi(\lambda)$ 所在的区域,来确定 $\varphi(\lambda)$ 的极值。典型方法为黄金分割法。

1. 插值法

求最优步长 λ 的实质是求单变量函数 $f(\lambda)$ 在某一区间 $\lambda_a \le \lambda \le \lambda_b$ 中的极小值,即

$$\min f(\lambda) \qquad \lambda_a \le \lambda \le \lambda_b \tag{10.3.2}$$

所谓插值法,就是利用常用的插值方法,用多项式来逼近函数 $f(\lambda)$,并建立一个便于计算和处理的近似表达式 $\varphi(\lambda)$,$\varphi(\lambda)$ 称为插值多项式。此多项式的极值点,就是原来函数 $f(\lambda)$ 的极值点的近似值。常用的插值多项式为二次或三次,分别称为二次插值方法和三次插值方法。

(1) 二次插值方法

如果已知函数 $f(\lambda)$ 在区间中的三个点 $\lambda_1 < \lambda_2 < \lambda_3$ 的函数值为 $f(\lambda_1)$, $f(\lambda_2)$ 和 $f(\lambda_3)$, 则可通过这三点 $(\lambda_1, f(\lambda_1))$, $(\lambda_2, f(\lambda_2))$ 和 $(\lambda_3, f(\lambda_3))$ 作一条抛物线, 并用此抛物线来逼近函数 $f(\lambda)$。设这个多项式为

$$\varphi(\lambda) = a_0 + a_1\lambda + a_2\lambda^2 \tag{10.3.3}$$

则

$$\left.\frac{d\varphi(\lambda)}{d\lambda}\right|_{\lambda=\lambda^*} = a_1 + 2a_2\lambda^* = 0$$

于是有

$$\lambda^* = -\frac{a_1}{2a_2} \tag{10.3.4}$$

这就是函数 $\varphi(\lambda)$ 的极小点, 也是函数 $f(\lambda)$ 的近似极小点。然后由已知的三个点值和函数值, 确定系数 a_1 和 a_2 的值。由于

$$\varphi(\lambda_1) = a_0 + a_1\lambda_1 + a_2\lambda_1^2 = f(\lambda_1)$$
$$\varphi(\lambda_2) = a_0 + a_1\lambda_2 + a_2\lambda_2^2 = f(\lambda_2)$$
$$\varphi(\lambda_3) = a_0 + a_1\lambda_3 + a_2\lambda_3^2 = f(\lambda_3)$$

解这个方程组求出 a_1, a_2, 代入 (10.3.4) 式得

$$\lambda^* = \frac{-\begin{vmatrix} 1 & f(\lambda_1) & \lambda_1^2 \\ 1 & f(\lambda_2) & \lambda_2^2 \\ 1 & f(\lambda_3) & \lambda_3^2 \end{vmatrix}}{2\begin{vmatrix} 1 & \lambda_1 & f(\lambda_1) \\ 1 & \lambda_2 & f(\lambda_2) \\ 1 & \lambda_3 & f(\lambda_3) \end{vmatrix}}$$

即求出极小点 λ^* 的值。实际应用时, 不直接采用一次抛物线逼近得到的 λ^* 作为最优步长, 而是要进行迭代。在迭代过程中, 将最优解 (λ^*, φ^*) 取代原三个点 (λ_1, φ_1), (λ_2, φ_2) 和 (λ_3, φ_3) 中最坏的一个点, 构成新的三个点。再通过这三个点重新进行抛物线逼近, 再次求得最优解。如果反复迭代, 直到相邻两次解的差足够小, 满足误差要求, 则认为一维搜索迭代收敛。收敛后的最优解 λ^* 即为最终最优解。

二次插值方法不必求函数的导数, 只需计算函数在区间内的三个点的函数值, 十分简单。

(2) 三次插值方法

如果函数 $f(\lambda)$ 在给定区间 $[\lambda_1, \lambda_3]$ 之间有极小点, 则 $f(\lambda)$ 必然满足

$$f'(\lambda_1) < 0$$
$$f'(\lambda_3) > 0$$

式中 $f'(\lambda_1)$ 和 $f'(\lambda_3)$ 分别代表 $f(\lambda)$ 在 λ_1 和 λ_3 点的一阶导数。设三次插值多项式为

$$\varphi(\lambda) = a_3(\lambda - \lambda_1)^3 + a_2(\lambda - \lambda_1)^2 + a_1(\lambda - \lambda_1) + a_0 \tag{10.3.5}$$

$\varphi(\lambda)$ 有极小值的充要条件是

$$\varphi'(\lambda^*) = 0$$
$$\varphi''(\lambda^*) > 0$$

即

$$\begin{aligned}\varphi'(\lambda^*) &= 3a_3(\lambda^* - \lambda_k)^2 + 2a_2(\lambda^* - \lambda_k) + a_1 = 0\\ \varphi''(\lambda^*) &= 6a_3(\lambda^* - \lambda_k) + 2a_2 > 0\end{aligned} \tag{10.3.6}$$

由此求出极小点 λ^*:

$$(\lambda^* - \lambda_k) = \frac{-a_1}{a_2 + \sqrt{a_2^2 - 3a_3a_1}} \tag{10.3.7}$$

为了求得系数 a_1, a_2 和 a_3, 可将 4 个已知条件, 即

$$\varphi(\lambda_k) = f(\lambda_k)$$
$$\varphi(\lambda_h) = f(\lambda_h)$$
$$\varphi'(\lambda_k) = f'(\lambda_k)$$
$$\varphi'(\lambda_h) = f'(\lambda_h)$$

代入式(10.3.5)和式(10.3.6), 得

$$\begin{aligned}&a_0 = f(\lambda_k)\\ &a_3(\lambda_h - \lambda_k)^3 + a_2(\lambda_h - \lambda_k)^2 + a_1(\lambda_h - \lambda_k) + a_0 = f(\lambda_h)\\ &a_1 = \varphi'(\lambda_k)\\ &3a_3(\lambda_h - \lambda_k)^2 + 2a_2(\lambda_h - \lambda_k) + a_1 = f'(\lambda_h)\end{aligned} \tag{10.3.8}$$

解方程组(10.3.8), 可求得插值多项式 $\varphi(\lambda)$ 的四个系数 a_0, a_1, a_2 和 a_3。再代入(10.3.7)式, 就可求出极小值 λ^*。然后经过迭代, 收敛于 $\varphi(\lambda)$ 的真正最优解。

三次插值方法需要计算函数的导数, 略为复杂, 但其收敛性较好。

2. 黄金分割法

黄金分割方法的主导思想是: 设 $f(\lambda)$ 是区间$[\lambda_l, \lambda_h]$中的单峰函数, 在区间$[\lambda_l, \lambda_h]$中任取两点 λ_1 和 λ_2, 且 $\lambda_1 < \lambda_2$。计算这两个点的函数值 $f(\lambda_1)$ 和 $f(\lambda_2)$, 并加以比较, 再按下列两种情况来缩短区间。

1) 如果 $f(\lambda_1) \leq f(\lambda_2)$, 则极小点必在 λ_l 和 λ_2 之间, 搜索区间可缩短为$[\lambda_l, \lambda_2]$。

2) 如果 $f(\lambda_1) > f(\lambda_2)$, 则极小点必在 λ_1 和 λ_h 之间, 搜索区间可缩短为$[\lambda_1, \lambda_h]$。

然后将缩短后的新区间记为$[\lambda_l', \lambda_h']$, 在$[\lambda_l', \lambda_h']$中同样再任取两点 λ_1'和 λ_2', 并通过比较函数值 $f(\lambda_1')$和 $f(\lambda_2')$, 获得进一步缩小的新区间$[\lambda_l'', \lambda_h'']$。如此不断反复, 即可得到近似极小点。

算法的关键是如何选择区间中两点 $\lambda_1^{(n)}$, $\lambda_2^{(n)}$ 的位置, 使得搜索区间缩小得尽可能快。黄金分割法是用黄金分割率 0.618 来确定中间点的值, 并使搜索区间长度每次都均匀地收缩到原来的 0.618 倍, 故这种算法也称为 0.618 法。

黄金分割方法的计算步骤(参考图 10.3.1)如下。

(1) 首次在初始区间$[\lambda_l, \lambda_h]$中选取两点 λ_1 和 λ_2;

$$\lambda_1 = \lambda_l + 0.618(\lambda_h - \lambda_l)$$
$$\lambda_2 = \lambda_h + 0.618(\lambda_h - \lambda_l)$$

(2) 计算函数值 $f(\lambda_1)$ 和 $f(\lambda_2)$;

(3) 比较 $f(\lambda_1)$ 和 $f(\lambda_2)$, 如果 $f(\lambda_1) \leq f(\lambda_2)$, 则把搜索区间缩小至 $[\lambda, \lambda_2]$, 即

$$\lambda' = \lambda, \quad \lambda_2' = \lambda_2$$

并在这个新区间 $[\lambda', \lambda_2']$ 重新选两点:

$$\lambda_2 = \lambda_1' + 0.618(\lambda_2' - \lambda_1')$$

$$\lambda_1 = \lambda_2$$

由于 λ_1 与 λ_2 重合, 则 $f(\lambda_1') = f(\lambda_2)$, 因此只需再计算一次函数值 $f(\lambda_2')$。相反, 如果 $f(\lambda_1) > f(\lambda_2)$, 则搜索区间缩小到 $[\lambda_1, \lambda_2]$, 即

$$\lambda' = \lambda_1, \quad \lambda_2' = \lambda_2$$

再在新区间中选取两点为

$$\lambda_1 = \lambda_2$$

$$\lambda_2 = \lambda_1' + 0.618(\lambda_2' - \lambda_1')$$

图 10.3.1 黄金分割算法

由于 $f(\lambda_1') = f(\lambda_2)$, 同样仅需计算 $f(\lambda_2')$。

(4) 比较 $f(\lambda_1')$ 和 $f(\lambda_2')$, 重复上述过程。如此反复迭代, 直到缩小后的区间 $[\lambda_1^{(n)}, \lambda_2^{(n)}]$ 满足 $|\lambda_2^{(n)} - \lambda_1^{(n)}| \leq \varepsilon$ 为止。ε 为预先给定的误差要求。

(5) 得到极小值的近似值:

$$\lambda^* = \frac{\lambda_1^{(n)} + \lambda_2^{(n)}}{2}$$

由上述算法步骤可得出其极小值误差为 $(0.168)^n/2$。

10.4 多变量函数优化

多变量函数优化主要解决如何确定搜索方向的问题。由于选取搜索方向 S^k 的方法不同, 就产生了不同算法。本节介绍几种常用的数值方法: 最速下降法、牛顿法、共轭梯度法以及变尺度法等。这些方法都属于梯度法, 需要计算目标函数的一阶导数或高阶导数。对求梯度较为困难的优化问题, 可采用不求导数的单纯形法, 我们在本节最后予以简单介绍。

1. 最速下降法

最速下降法是梯度法中最基本的一种算法。由于目标函数的负梯度方向是函数下降最快的方向, 故称为最速下降法。最速下降法的寻优过程简述如下。

给定目标函数 $F(P)$, P 是 n 维向量。设 P^k 和 P^{k+1} 表示第 k 步和第 $k+1$ 步迭代的 P 值, 并定义

$$\Delta P^k = P^{k+1} - P^k \tag{10.4.1}$$

ΔP^k 表示两个向量 P^{k+1} 和 P^k 之差的量值和方向。因在迭代过程中目标函数是不断下降的, 因此有

$$F(P^{k+1}) = F(P^k + \Delta P^k) < F(P^k)$$

在 P^k 附近,对 $F(P^k+\Delta P^k)$ 进行台劳级数展开,略去高次项,得

$$F(P^k+\Delta P^k)=F(P^k)+[\grad F(P^k)]^T\cdot\Delta P^k \tag{10.4.2}$$

则有

$$[\grad F(P^k)]^T\cdot\Delta P^k<0 \tag{10.4.3}$$

即要求式(10.4.3)表示的点积是负值,这相当于参数增量向量 ΔP^k 的方向与梯度向量 $F(P^k)$ 的方向相反。

设 S^k 是与 ΔP^k 相同方向的单位向量,即

$$S^k=\frac{\Delta P^k}{\|\Delta P^k\|}$$

式中 $\|\Delta P^k\|$ 是个正实数,可用 λ(称之为步长)表示,则

$$\Delta P^k=S^k\cdot\lambda \tag{10.4.4}$$

将式(10.4.4)代入式(10.4.3),得

$$\lambda[\grad F(P^k)]^T\cdot S^k<0 \tag{10.4.5}$$

由于 $\lambda>0$,故有

$$[\grad F(P^k)]^T\cdot S^k<0$$

又由

$$[\grad F(P^k)]^T\cdot S^k=\|\grad F(P^k)\|\,\|S^k\|\cos\theta$$

可知,当 $\theta=180°$ 时,得负的最大值。也就是说,如果选此梯度负方向作为 S^k 的方向,可以使目标函数 $F(P)$ 的下降最快,此时

$$S^k=-\frac{\grad F(P^k)}{\|\grad F(P^k)\|} \tag{10.4.6}$$

最速下降法的优化过程如下:

(1) 给定允许误差 ε($\varepsilon>0$),令迭代次数为 k,首次迭代 k=0,并选取迭代初值 P^0。

(2) 计算梯度向量 $\grad F(P^k)$。若 $\|\grad F(P^k)\|<\varepsilon$ 则 P^k 为最优解,迭代结束;否则转(3)。

(3) 计算搜索方向 S^k:

$$S^k=-\frac{\grad F(P^k)}{\|\grad F(P^k)\|}$$

(4) 负梯度方向确定后,设沿此方向前进的最佳步长为 λ,要用一维搜索方法求自变量为 λ 的一元函数的最小值 λ,即求解

$$F(P^k+\lambda S^k)=\min_{\lambda\geq 0}F(P^k+\lambda S^k)$$

(5) 令

$$P^{k+1}=P^k+\lambda S^k$$

又令 k=k+1,转(2)。

最速下降法的优化过程示意图如图 10.4.1 所示。

例 10.4.1　设目标函数 $F(P)=p_1^2+p_1p_2+p_2^2$,用最速下降法求极小值。

(1) 任选初始值

$$P^0=[p_1^0\quad p_2^0]^T=[2\quad -3]^T$$

给定收敛误差 $\varepsilon = 0.001$。

(2) 计算梯度值$\nabla F(P^k)$:

$$\nabla F(P) = \left[\frac{\partial F(P)}{\partial p_1} \quad \frac{\partial F(P)}{\partial p_2}\right]^T$$

$$= [2p_1 + p_2 \quad p_1 + 2p_2]^T$$

于是

$$\nabla F(p^0) = [1 \quad -4]^T, \quad F(P^0) = 7$$

即

$$\| \nabla F(P^0) \| = \sqrt{1^2 + (4)^2}$$

$$= \sqrt{17} = 4.123$$

显然不满足收敛条件。

图 10.4.1 最速下降法示意图

(3) 计算搜索方向 S^k:

由

$$S^k = \frac{-\nabla F(P^k)}{\| \nabla F(P^k) \|}$$

得

$$S^0 = \frac{-[-1 \quad -4]^T}{\sqrt{17}} = -\left[\frac{1}{\sqrt{17}} \quad \frac{4}{\sqrt{17}}\right]^T = \begin{bmatrix} -0.2425 \\ 0.9701 \end{bmatrix}$$

(4) 用一维搜索方法确定最佳步长 λ:

$$F(P^k + \lambda S^k) = \min_{\lambda \geq 0} F(P^k + \lambda S^k)$$

表 10.4.1 是采用黄金分割法在区间[0.5, 3.0]内进行一维搜索, 对步长 λ 进行优化的结果。

表 10.4.1 采用黄金分割法确定最佳步长的迭代过程

λ	0.5	1.455	2.045	2.410	2.549	2.635	2.775	3.0
p_1^1		1.647	1.504	1.415	1.382	1.361	1.327	
p_2^1		-1.588	-1.016	-0.662	-0.527	-0.444	-0.308	
$F(P^1)$		2.700	1.770	1.505	1.460	1.445	1.447	

根据此表, 取 $\lambda^* = \frac{2.635 + 2.775}{2} \approx 2.7$ 为最佳步长。

(5) 计算下一个迭代点 P^1 数值:

$$P^1 = P^0 + \lambda S^0 = [1.345 \quad -0.381]^T$$

则函数值 $F(P^1) = 1.442$。

由于 $F(P^0) = 7$, 可以看出 $F(P^1) < F(P^0)$, 函数值是减小的。转步骤(2)继续迭代。直到获得 $F(P)$ 的极小值为止。

最速下降法方法简单, 在迭代初期收敛速度较快。它的缺点是在极小点附近收敛很慢。这是因为大多数目标函数在极小点附近都接近于二次函数, 而最速下降法的台劳展开式只取了一阶项。因此, 如果我们能将展开式取到二阶, 则会使算法收敛性得到改善, 这就

是二阶梯度法的基本思想。我们下面介绍的牛顿法、变尺度法和共轭梯度法都属于二阶梯度法。

2. 牛顿法

如果目标函数 $F(P)$ 具有一阶和二阶偏导数，我们在第 k 次迭代点 P^k 附近，取 $F(P^k + \Delta P^k)$ 的二阶台劳展开式逼近 $F(P)$，即

$$F(P^k + \Delta P^k) \approx F(P^k) + [\nabla F(P^k)]^T \Delta P^k + \frac{1}{2}(\Delta P^k)^T H^k \Delta P^k \qquad (10.4.7)$$

式中 H 是 $F(P)$ 在 P^k 点的二阶导数矩阵，即海森(Hessian)矩阵。

求式(10.4.7)的梯度，并令其等于零，即

$$\nabla F(P^k) + H^k \Delta P^k = 0$$

或写成

$$P^{k+1} = P^k - [H^k]^{-1} \nabla F(P^k) \qquad (10.4.8)$$

如果目标函数是严格的二次函数，H 是个常数矩阵，用这种方法只需迭代一次就可以达到极小点。而式(10.4.7)仅仅是 $F(P)$ 的近似表达式，即 $F(P)$ 不是严格的二次函数，因此需要进行多次迭代。可以从 P^k 点出发，取搜索方向为

$$S^k = -[H^k]^{-1} \nabla F(P^k) \qquad (10.4.9)$$

沿此方向进行一维搜索，并用迭代公式

$$P^{k+1} = P^k - \lambda [H^k]^{-1} \nabla F(P^k) \qquad (10.4.10)$$

进行迭代运算，直到找到极小点为止。

牛顿法的优点是利用了函数的二次导数信息，收敛速度大大地加快了。它的缺点是每次迭代都要计算二阶导数矩阵的逆矩阵，当 P 的维数较高时，计算工作量很大。我们希望不直接求 $[H^k]^{-1}$，而又能达到收敛较快的效果。这就是我们下面要介绍的变尺度法。

3. 变尺度法

变尺度法的原理是：用一阶偏导数组合成一个与 H^k 同阶的矩阵 A^k，以 A^k 近似表示海森逆矩阵 $[H^k]^{-1}$，从而避免了求二阶导数和求逆的困难。此法又叫拟牛顿法。

由于构造方向矩阵 A 的形式有很多种，不同的形式就构成不同的变尺度法。我们这里只介绍最常用的 DFP 法(60 年代由 Davidon，Fletcher 和 Powell 提出)。

DFP 法的迭代公式是

$$P^{k+1} = P^k - \lambda A^k \nabla F(P^k) \qquad (10.4.11)$$

方向矩阵 A 的表达式为

$$A^{k+1} = A^k + \frac{\Delta P^k (\Delta P^k)^T}{(\Delta P^k)^T Y^k} - \frac{A^k Y^k (A^k Y^k)^T}{(Y^k)^T A^k Y^k} \qquad (10.4.12)$$

式中

$$Y^k = \nabla F(P^{k+1}) - \nabla F(P^k)$$

变尺度算法在首次迭代时，若令迭代公式(10.4.11)中的 A^k 等于单位阵，即为最速下降法；在以后的迭代中按式(10.4.12)计算海森逆矩阵。

变尺度法的算法步骤如下：

(1) $k=0$，给定初值 P^0 和误差要求 ε 并取 $A^0 = I$。

(2) 确定一维搜索的方向向量：

$$S^{(k)} = -A^k \nabla F(P^k)$$

(3) 进行一维搜索:

$$F(P^{k+1}) = F(P^k + \lambda S^k) = \min_{\lambda \geq 0} F(P^k + \lambda S^k)$$

(4) 求梯度向量$\nabla F(P^{k+1})$,并校验是否满足

$$\| \nabla F(P^{k+1}) \| \leq \varepsilon$$

若满足,表示已求得最小点,优化结束。否则继续下一步。

(5) 求梯度向量的差向量 Y^k:

$$Y^k = \nabla F(P^{k+1}) - \nabla F(P^k)$$

(6) 求矩阵 A^{k+1}:

$$A^{k+1} = A^k + \frac{\Delta P^k (\Delta P^k)^T}{(\Delta P^k)^T Y^k} - \frac{A^k Y^k (A^k Y^k)^T}{(Y^k)^T A^k Y^k}$$

(7) 令 $k = k+1$,返回步骤(2),继续迭代运算。

变尺度法克服了牛顿法必须直接计算海森矩阵逆矩阵的缺点,而且兼有牛顿法和最速下降法二者的优点,能达到收敛速度较快的效果。

4. 共轭梯度法

共轭梯度法是一种比较有效的寻优方法。与 DFP 方法相比较,它可以不必计算海森矩阵,从而节省了计算量和存储量,对高维问题有实际意义。

首先介绍共轭方向概念,然后介绍如何确定共轭方向。

(1) 共轭方向

对于一个 $n \times n$ 维的对称正定矩阵 A,如果有一组向量$\{S^k\}$(其中 $k = 1, 2, \cdots, n$),当它们满足如下关系:

$$(S^i)^T A S^j = 0 \quad i \neq j; \quad i, j = 1, 2, \cdots, n \tag{10.4.13}$$

就称这组向量是 A 的共轭向量。

一组 A 的共轭向量 S^k 有如下特点:

1) 如果 A 是正定矩阵,则$\{S^k\}$中各向量 $S^1, S^2, \cdots$是互相线性独立的。

2) 如果 A 是单位矩阵,则$\{S^k\}$为一组正交向量。所以正交是共轭的特殊情况,共轭是正交概念的推广。

3) 在寻优过程中,可能产生某一个梯度向量$\nabla F(P^k)$与 $S^1, S^2, \cdots, S^{k-1}$都正交。这时$\nabla F(P^k)$必为零,即

$$\nabla F(P^k) = 0$$

也就是说 P^k 为最优解。

(2) 确定共轭方向

一般说来,大多数目标函数在极小点附近是近似于二次函数的,所以我们以二次函数为例,讨论共轭搜索方向的确定方法。

作为一种迭代算法,我们希望共轭方向在迭代过程中逐次产生。因此在迭代的第一步,取负梯度方向作为搜索方向,即

$$S^0 = -\nabla F(P^0) \tag{10.4.14}$$

在以后各步中,搜索方向都要对该步的负梯度方向作一修正,即将负梯度方向旋转一个角

度:

$$S^{1} = -\nabla F(P^{1}) + \beta S^{0} \tag{10.4.15}$$

以保证 S^{1} 与 S^{0} 关于 A 互为共轭, 即

$$(S^{0})^{T} A S^{1} = 0 \tag{10.4.16}$$

式(10.4.15)中, β 称为共轭系数。

由(10.4.7)式可知,

$$\nabla F(P^{1}) = \nabla F(P^{0}) + A\Delta P^{0}$$

或写为

$$\nabla F(P^{1}) - \nabla F(P^{0}) = A(P^{1} - P^{0}) = A\lambda S^{0}$$

$$\lambda S^{0} = A^{-1}[\nabla F(P^{1}) - \nabla F(P^{0})] \tag{10.4.17}$$

将(10.4.17)、(10.4.15)式代入(10.4.16)式, 得

$$[\nabla F(P^{1}) - \nabla F(P^{0})]^{T}[-\nabla F(P^{1}) + \beta S^{0}] = 0 \tag{10.4.18}$$

已知 $\nabla F(P^{1})$ 与 S^{0} 正交, 即 $[\nabla F(P^{1})]^{T} S^{0} = 0$, 而且 $\nabla F(P^{1})$ 与 $\nabla F(P^{0})$ 正交, 即 $[\nabla F(P^{0})]^{T} \nabla F(P^{1}) = 0$, 故由(10.4.18)式可得

$$\| \nabla F(P^{1}) \|^{2} = \beta \| \nabla F(P^{0}) \|^{2}$$

或

$$\beta = \frac{\| \nabla F(P^{1}) \|^{2}}{\| \nabla F(P^{0}) \|^{2}} \tag{10.4.19}$$

由此推广, 可得第 $k+1$ 步的共轭系数为

$$\beta = \frac{\| \nabla F(P^{k+1}) \|^{2}}{\| \nabla F(P^{k}) \|^{2}} \tag{10.4.20}$$

则

$$S^{k+1} = -\nabla F(P^{k+1}) + \beta S^{k} \tag{10.4.21}$$

由此看出, 可以直接从函数 $F(P)$ 的梯度计算 β, 而不必知道矩阵 A, 这样在求共轭方向时就避免了计算二阶偏导数矩阵 A 的麻烦。

例 10.4.2　目标函数为 $F(P) = \frac{3}{2}p_1^2 + \frac{1}{2}p_2^2 - p_1 p_2 - 2p_2$, 用共轭梯度法求其极小值。

(1) 设初值 $P^{0} = [-2 \quad 4]^{T}$, 误差 $\varepsilon = 0.001$, 计算梯度向量:

$$\nabla F(P^{0}) = [3p_1 - p_2 - 2 \quad p_2 - p_1]^{T} = [-12 \quad 6]^{T}$$

首次取负梯度方向为搜索方向, 即

$$S^{0} = -\nabla F(P^{0}) = [12 \quad -6]^{T}$$

(2) 求 $\min_{\lambda} F(P^{0} + \lambda S^{0})$:

令 $\frac{F}{\lambda} = 0$, 得 $\lambda^{*} = \lambda = \frac{5}{17}$。

(3) 计算 P^{1} 时的梯度向量:

$$P^{1} = P^{0} + \lambda S^{0} = \begin{matrix} -2 \\ 4 \end{matrix} + \frac{5}{17}\begin{matrix} 12 \\ -6 \end{matrix} = \frac{1}{17}\begin{matrix} 26 \\ 38 \end{matrix}$$

$$\nabla F(P^{1}) = \begin{matrix} \frac{6}{17} & \frac{12}{17} \end{matrix}^{T}$$

判断误差:

$$\| \nabla F(P^1) \| = \sqrt{\left(\frac{6}{17}\right)^2 + \left(\frac{12}{17}\right)^2} = 0.789 > \varepsilon$$

(4) 计算共轭系数和共轭方向:

$$\beta = \frac{\| \nabla F(P^1) \|^2}{\| \nabla F(P^0) \|^2} = \frac{1}{289}$$

$$S^1 = -\nabla F(P^1) + \beta S^0 = -\begin{bmatrix} \frac{6}{17} \\ \frac{12}{17} \end{bmatrix}^T + \frac{1}{289}[12 \quad -6]^T$$

$$= \frac{1}{289}[-90 \quad -210]^T$$

(5) 返回步骤(2),求 $\min_{\lambda} F(P^1 + \lambda S^1)$:

令 $\frac{F}{\lambda} = 0$,得 $\lambda^* = \lambda = \frac{17}{10}$。再计算 P^2,有

$$P^2 = P^1 + \lambda S^1 = \begin{bmatrix} \frac{1}{17} \\ \frac{26}{38} \end{bmatrix} + \frac{17}{10}\frac{1}{289}\begin{bmatrix} -90 \\ -210 \end{bmatrix} = \begin{bmatrix} 1 \\ 1 \end{bmatrix}$$

这时有 $\nabla F(P^2) = 0$,故 P^2 为最优点。

采用共轭梯度法时,一般对 n 维变量只进行 n 次(或 n+1 次)循环迭代。超过此数后,则将此时所得近似最小点 P^n 作为第二轮循环的初值点 P^0,重新开始第二轮循环。

共轭梯度法实际上是对最速下降法的一种改进算法,使搜索方向从负梯度方向转了某个角度,它综合利用了本步梯度与过去各点梯度的信息,因此在极小点附近收敛速度较快。而且不必计算海森矩阵,计算量也大大降低。

5. 单纯形法

前面介绍的几种优化方法属于梯度优化方法,它们需要对目标函数求导,甚至要用到二阶偏导数才能判断搜索方向。在电路优化中,有时目标函数十分复杂,求解偏导数很困难,这时最好采用直接搜索法。这种方法可以不求导数,只需计算目标函数在若干点的函数值,再进行比较就能判断搜索方向。

单纯形法是一种多维直接搜索优化方法。所谓单纯形是指在一定空间中,由直线构成的最简单的图形。在二维空间中单纯形是三角形,三维空间的单纯形是有四个顶点的四面体,N 维空间的单纯形是 N+1 个顶点的几何形体。单纯形法多维搜索寻优是利用单纯形的顶点,计算其函数,按一定规律进行探测性搜索。对搜索区间内单纯形顶点进行比较,判断目标函数的变化趋势,确定有利的搜索方向和步长。显然,以单纯形顶点作为目标函数值判断的依据,计算点数最少。

下面以二元函数为例介绍单纯形法的基本原理。

设有一个二元目标函数 F(P)。在 p_1、p_2 平面上选择任意三个点构成一个三角形(即二维单纯形),并计算和比较这三个点的函数值。函数值最大的点是最差点,记作 P_H;函数值最小的点为最好点,记作 P_L;函数值介于 P_H 和 P_L 之间的点为中间点,记作 P_G;如图 10.4.3 所示。

根据 $F(P_H)$,$F(P_G)$和 $F(P_L)$三个函数值可以大略估计出目标函数的下降方向,取该方向为搜索方向。显然取最差点 P_H 的反对称方向为搜索方向,目标函数会有所改善。取

图 10.4.2　单纯形

(a) 二维单纯形;　　(b) 三维单纯形

P_G 和 P_L 的联线中点为 P_C, 从 P_H 指向 P_C 的方向就是搜索方向, 在此方向上选取反射点 P_R, 使

$$P_R = P_C + \gamma(P_C - P_H) \tag{10.4.22}$$

其中 γ 为反射系数, 根据经验一般取 $\gamma = 1$, 故

$$P_R = 2P_C - P_H \tag{10.4.23}$$

将求得的 $F(P_R)$ 与其它三点函数比较, 有下面几种可能性:

(1) $F(P_R) < F(P_L)$, 即比原来最好点的目标函数值还小, 说明搜索方向正确, 可以沿此方向试搜索更远一些, 故取扩展点 P_E (参见图 10.4.3):

图 10.4.3　单纯形法示意图

$$P_E = P_C + \alpha(P_C - P_H) \tag{10.4.24}$$

α 为扩展系数, 一般取 $\alpha > 1$, 例如 $\alpha = 1.2 \sim 2$。

(2) $F(P_R) > F(P_G)$, 即 P_R 点函数值比原单纯形中的中间值还差, 表示 P_R 取得太远, 应返回一些, 取压缩点 P_S:

$$P_S = P_C + \beta(P_R - P_C) \tag{10.4.25}$$

β 为压缩系数, 一般 $0 < \beta < 1$, 可取 $\beta = 0.5$ 或 $0.25 \sim 0.75$。

(3) $F(P_R) > F(P_H)$, 即 P_R 点函数值比原来最差点 P_H 还要差, 则应压缩更多, 新点 P_S 应位于 P_H 和 P_C 之间, 即

$$P_S' = P_C - \beta(P_C - P_H) = P_C + \beta(P_H - P_C) \tag{10.4.26}$$

这样就完成了第一次搜索过程。

下一步是将 P_H 去掉, 得到新的二维单纯形。再对新单纯形的三个顶点 P_G, P_R, P_L 计算并比较其函数值。如果 $F(P_G) > F(P_R) > F(P_L)$, 则应求 P_G 点的反射点 P_K。再对形成

的新单纯形三顶点 P_R, P_L, P_K 做计算比较，直到最差点与最好点的函数值的差别小于给定的误差为止。

上述原则同样适用于 n 维情况，只是 n 维单纯形有 n+1 个顶点，计算工作量大。n 维情况下单纯形法的步骤如下：

(1) 给定初始参数：初始点 P_0，变量数 n，步长 h，扩展因子 α 压缩因子 β 最大允许搜索次数 k 等等。

(2) 根据步长计算出 n+1 个顶点：

$$P_i = P_0 + h\delta \quad (i = 1, 2, \cdots, n)$$

$$P_{n+1} = P_0$$

(3) 计算 n+1 个顶点的函数值，并从中选出最大、次大和最小的三个点：

$Y_H = F(P_H)$ ——函数最大点值；

$Y_L = F(P_L)$ ——函数最小点值；

$Y_G = F(P_G)$ ——函数次大点值。

(4) 判收敛：满足

$$\left| Y_L^{k+1} - Y_L^k \right| < \varepsilon$$

时为收敛，否则继续搜索。

(5) 计算中心点(或重心点) P_C 和反射点 P_R：

$$P_C = \frac{1}{n} \sum_{i=1}^{n+1} P_i - P_H$$

$$P_R = 2P_C - P_H \quad (\gamma = 1)$$

(6) 若 $F(P_R) \geq F(P_G)$，进行压缩，令

$$P_S = P_C + \beta(P_R - P_C) \quad 0 < \beta < 1$$

转(8)；若 $F(P_R) < F(P_G)$，转(7)。

(7) 进行扩展，令

$$P_E = P_C + \alpha(P_C - P_H) \quad \alpha > 1$$

如果 $F(P_E) < F(P_R)$，令 $P_S = P_E$。否则令 $P_S = P_R$。

(8) 若 $F(P_S) < F(P_G)$，将 P_H 换成 P_S，并和其它 n 个点构成新的单纯形，重新确定 Y_H, Y_L, Y_G，并返回步骤(4)继续迭代。否则转(9)。

(9) 进行缩小单纯形，令

$P_i = (P_i + P_L)/2$，P_i 表示除 P_L 点外的各点。然后重新开始，直到满足收敛条件为止。

总之，单纯形法不必计算梯度，也不用按搜索方向进行一维寻优或按固定步长寻优，它是依靠不断地比较函数值来判断函数下降方向，用一个新的较好的点来代替原来单纯形中较差的点。因此单纯形法的初始点和步长的选取十分重要。

10.5 有约束优化方法

前面介绍的各种优化方法对变量值没有任何限制，故叫无约束最优化。在实际电路优

化设计中,待优化的变量往往都受一定条件的限制。比如,电路中的元器件参数:电阻、电容、电感或半导体器件参数都必须为正值;当以放大器的增益带宽作为目标函数时,其功耗就可能是一种约束条件。在优化问题中,除了使目标函数最小之外,还需满足一些约束条件,我们称之为有约束的优化问题。

约束条件可分为等式约束与不等式约束两大类。等式约束上各点称为可行解,因此等式约束曲线表示可行解域。一般不等式约束在自变量空间构成一个可行域,在这个域内的解都是可行的,称为可行解。可行解的数目有无限多个,其中必有一个是最优解。

例 10.5.1　求 $\min F(p)=(p-a)^2+b$,

(1) 等式约束　$p=c$;

(2) 不等式约束　$p\geq c$,设 $c<a$。

这是单变量优化问题。

解:(1) 等式约束 $p=c$ 为一条直线,这时 $p=c$ 为可行域,唯一解即为 $p^*=c$。如图 10.5.1(a)。

图 10.5.1　有约束优化

(a) 等式约束;　(b) 不等式约束

(2) 不等式约束 $p\geq c$(设 $c<a$)。可行域在直线 $p=c$ 的右边,如图 10.5.1(b)所示。这时最优解和无约束问题一样,$p^*=a$,$f(p^*)=b$。但如果不等式约束为 $p\leq c$,$c<a$,则可行域在直线 $p=c$ 的左边,这时最优解为 $p^*=c$,即最优解在约束边界上。

由此例可以看出,对同一目标函数,有约束和无约束,或约束条件有所不同,其优化结果往往相差很多。

有约束优化方法有多种,我们这里只介绍罚函数法,它是实际中应用最广泛而又较为简单、有效的一种数值优化方法。罚函数方法的实质是将有约束优化问题通过惩罚因子的选择变为一系列求罚函数的极小值问题,从而将原问题转化为求解一系列无约束极值问题。转化为无约束优化问题以后,就可以用前面所介绍的任何一种最优化方法来进行设计了。

1. 等式约束的罚函数法

设等式约束的优化为

$$\begin{aligned} &\min F(P) \\ &g_i(P) = 0 \quad i = 1, 2, \cdots, m \end{aligned} \tag{10.5.1}$$

罚函数法是将原目标函数 $F(P)$ 增广为一个新函数，称为罚函数 Φ，其形式为

$$\Phi(P, M_k) = F(P) + M_k \sum_{i=1}^{m} [g_i(P)]^2 \tag{10.5.2}$$

其中 M_k 是个很大的正数，称为罚因子或罚系数。当 $g_i(P) \neq 0$，即不满足约束等式时，罚函数 $\Phi(P, M_k)$ 的值将很大，不能收敛；只有当 $g_i(P) = 0$ 时，罚函数 Φ 才等于原函数 $F(P)$。所以罚函数法是把约束条件作为一个“惩罚项”加进原目标函数中，只有满足约束条件时目标函数才不受“惩罚”，仍按原目标函数进行优化。

$M_k \sum_{i=1}^{m} [g_i(P)]^2$ 称为惩罚项或约束函数，$\Phi(P, M_k)$ 是约束函数与目标函数的组合函数，求 $\Phi(P, M_k)$ 的极小值是一个无约束最优化问题：

$$\min \Phi(P, M_k) = \min F(P) + M_k \sum_{i=1}^{m} [g_i(P)]^2$$

它等价于求原函数在等式约束 $g_i(P) = 0$ 下的最优化问题。

$\Phi(P, M_k)$ 的最优点与 M_k 有关，记作 $P_k^* = P^*(M_k)$，每迭代一步，选择一个罚因子，M_k 表示第 k 步迭代时所取的罚因子，P_k^* 表示第 k 步迭代时所得无约束最优化问题 $\min \Phi(P, M_k)$ 的最优解。显然 M_k 应大于 M_{k-1}，即迭代过程中罚因子越取越大，迫使 $g_i(P)$ 趋近于零，才能使 $\Phi(P, M_k)$ 为极小，最后 P_k^* 收敛于 $F(P)$ 的最优解。

2. 不等式约束的罚函数法

设不等式约束优化问题为

$$\begin{aligned} &\min F(P) \\ &g_i(P) \leq 0 \quad i = 1, 2, \cdots, m \end{aligned} \tag{10.5.3}$$

如果不等式约束为 $g_i(P) \geq 0$，则可以将它变换为 ≤ 0 的形式：$-g_i(P) \leq 0$。

构造罚函数：

$$\Phi(P, M_k) = F(P) + M_k \sum_{i=1}^{m} \{\max[g_i(P), 0]\}^2 \tag{10.5.4}$$

式中 $\max[g_i(P), 0]$ 表示在 $g_i(P)$ 和 0 之间选择最大者，即

$$\max[g_i(P), 0] = \begin{cases} 0 & g_i(P) \leq 0 \\ g_i(P) & g_i(P) > 0 \end{cases} \tag{10.5.5}$$

当搜索点序列 $\{P^k\}$ 在可行解域内，则 $\max[g_i(P), 0] = 0$；如果 $\{P^k\}$ 在可行解域外，即这时没有满足约束条件，我们令 $\max[g_i(P), 0] = g_i(P)$，这时将不等式约束按等式约束处理，罚因子 M_k 起作用。因此，如果 $\{P^k\}$ 在可行解域外而又远离边界，则 $\Phi(P, M_k)$ 将会很大，罚因子的作用是不让 P^k 远离边界。

例 10.5.2　求解不等式约束优化问题

$$\begin{aligned} &\min F(P) = p_1^2 + 2p_2^2 \\ &g(P) = p_1 \geq 1 \end{aligned}$$

构造罚函数

$$\Phi(P, M_k) = p_1^2 + 2p_2^2 + M_k[\max(p_1 - 1, 0)]^2$$

即

$$\Phi(P, M_k) = \begin{cases} p_1^2 + 2p_2^2 + M_k(p_1 - 1)^2 \\ p_1^2 + 2p_2^2 \end{cases}$$

用解析法对上式求极值,即令其一阶导数等于零:

$$\frac{\Phi}{p_1} = 2p_1 + 2M_k(p_1 - 1) = 0$$

$$\frac{\Phi}{p_2} = 2p_2 = 0$$

得

$$p_1 = \frac{M_k}{1 + M_k}$$

$$p_2 = 0$$

当 $M_k \to \infty$ 时,p_1 的极值 $\to 1$,即 $p_1 = 1$,$p_2 = 0$。此时目标函数 $F(P) = 1$,这就是有约束的极小值。这种优化方法,当 $M_k \to \infty$ 时 $P_k^* = P^*(M_k)$ 是由非可行解域逐步向约束边界逼近收敛的,故称之为外罚函数法。

外罚函数法的具体迭代步骤如下:

(1) 设起始点 P^0,给定初始罚因子 M_1,且 $M_{k+1} = CM_k$,$C > 1$。令 $k = 1$。

(2) 应用任何一种无约束优化方法求罚函数 $\Phi(P, M_k)$ 的极小值 P_k^*。

(3) 检验 P_k^* 是否满足约束条件和优化收敛判据。若是,则 P_k^* 为最优解,迭代结束;否则继续下一步。

(4) 令 $k = k + 1$,转步骤(2)。

不等式约束的罚函数优化方法还有内罚函数法、混合罚函数法等等,在此就不一一列举了。

10.6 统计优化方法

1. 统计优化方法概述

在电路拓扑结构与元器件类型给定的条件下,电路的性能完全由元器件参数来确定。在电路 CAD 领域中,优化设计的主要目的是进行电路参数优化,从而使电路性能达到设计指标。它通常是求解一个确定性的数学规划问题,这个数学规划问题的目标函数是电路实际性能与给定的性能指标之差,通过优化设计使目标函数极小化。这也就是我们在前面几节介绍的若干种优化设计方法。采用这类方法进行电路设计,有可能求得的最优解并不是最好的电路设计方案。原因在于这种参数优化得出的设计方案可能不允许电路参数存在容差。当某些参数值产生微小变动时就可能引起电路性能很大的变化,从而偏离性能指标规定的要求。图 10.6.1 示意地画出了一维参数空间中的这种情况。图中 p 是电路参数,目标函数 $e(p)$ 是电路性能与要求指标之差的绝对值。当 $e(p) \leq \varepsilon$ 时,可以认为电路性能满足指标要求,电路合格。合格域为 $R_A = \{p \mid p_l \leq p \leq p_u\}$。显然,如果电路参数优化过程收敛,得到的最优解为 p^*。这样的设计方案若投入实际生产,电路参数稍稍偏离 p^*,电路性

能就会迅速变坏,使批量生产合格率很低。在电路实际批量生产制造过程中和使用过程中,由于各种不可控、不可测的随机因素影响,电路中的元器件参数的实际值相对于其设计值总存在一定的偏差。参数偏离设计值,电路性能也要偏离设计值;电路参数随机变化,使电路性能也产生随机波动。一旦这种波动超出了指标要求,电路就不能完成预期的功能。满足设计指标要求,能完成预期功能的电路在总生产量中所占的比例称为生产合格率。我们希望在保证电路合格率足够高,允许电路参数的随机波动尽可能大的情况下设计电路。这种考虑电路参数的容差及以随机性的电路优化问题,我们称之为统计优化问题,或统计电路设计。

图 10.6.1　最优解位于合格域边界

在电路的统计优化设计中,若电路参数分布的统计特性已知,我们希望设计电路参数的中心值,使电路性能满足设计要求,进而使生产合格率尽可能高。所以又称之为最优中心值设计问题。目前所提出的各种统计优化方法大致可分为两类:

(1) 确定性方法。这种方法试图用一确定性数学规划问题的序列逐步逼近最优中心值设计问题的解,称之为确定性统计电路设计方法。例如某种确定性方法可以对合格域边界做出某种描述,将参数中心选在合格域中心点,并在所确定的合格域内嵌入一个尽可能大的容差域,而使合格率达到最大。但这类方法对电路性能函数或合格域的性质有较多的要求,在实际电路中往往不能满足或得不到验证。而且随着参数空间维数的增加,算法的计算量和复杂程度迅速增加,因而不太适于处理较大规模的电路优化问题。

(2) 统计性方法。这种方法采用各种改进的蒙特卡罗(Monte-Carlo)方法估计电路合格率的数值或梯度值,据此来改变电路参数中心值,以提高合格率。这类方法编程简单,参数空间对算法的有效性和计算量几乎没有影响,对电路的性能函数及合格域的性质没有特殊要求,适合于大规模电路的优化设计。这种方法的缺点是在不同参数中心值上进行抽样点数很大的蒙特卡罗分析和电路模拟,因而计算量仍较大。

具体的统计优化算法有很多种。例如,以实现 100% 合格率为目的的最坏条件设计;在给定合格率的情况,求最大的容差域;在给定合格率的情况下,求最佳的电路性能等等。下面我们着重介绍统计优化方法中以合格率最大为目标的中心值优化方法。

2. 中心值优化的数学模型

首先介绍一些基本定义。

(1) 合格域 R_A

当给定电路的性能指标约束后,满足这个性能指标要求的元件参数空间,称为合格域(acceptable region),记作 R_A。电路元件参数在合格域内,则电路性能满足要求,即是合格的,否则是不合格的。合格域 R_A 的数学描述是

$$R_A = \{P \mid f_j^{min} \leq f_j(P) \leq f_j^{max}, j = 1, 2, \cdots, m\} \tag{10.6.1}$$

式中 $f_j(P)$ 表示第 j 项电路性能; f_j^{max}, f_j^{min} 分别表示第 j 项性能的上限约束和下限约束。

(2) 容差域 R_T

元件参数容差域的表达式为

$$R_T = \{P \mid p_i^0 - t_i \leq p_i \leq p_i^0 + t_i, i = 1, 2, \cdots, n\} \tag{10.6.2}$$

式中 p_i, p_i^0 分别表示第 i 个元件参数值及参数中心值; t_i 表示第 i 个元件参数允许的容差范围。对于均匀分布的元件参数,容差域是 n 维参数空间的一个超长方体。

图 10.6.2 用二维情况表示了合格域 R_A 与容差域 R_T 之间的关系。可以看出当元件参数中心值和容差域给定时,每个电路相当于容差域中的一个点; 当点落在合格域内时,电路是合格的,这组电路参数是可行的; 否则电路就不合格。

(3) 合格率

合格率的数学定义是

$$Y(P^0, t) \overset{def}{=} \psi(P \in R_A) = \int_{R_A} \Phi(P, P^0) dP \quad P, P^0 \in R^n \tag{10.6.3}$$

式中 $\Phi(P, P^0)$ 为参数的概率密度函数; P^0 为元件参数中心值, t 为各元件容差。

从图 10.6.2 中可以看出,当参数概率密度分布不变,容差不变的情况下,通过移动元件参数中心值(相当于图中容差域平移)就可提高合格率,即

$$\max_{P^0} Y(P^0, T) \tag{10.6.4}$$

图 10.6.2 最优中心值设计示意图

这就是中心值优化问题,也叫“最优中心值设计”(design centering)问题。

电路的合格域 R_A 边界一般很难确定。我们采用统计法的中心值优化方法是以蒙特卡罗分析为基础,它并不试图确定合格域的边界,而是对给定元件参数中心值附近的随机数进行蒙特卡罗分析,计算出电路合格率; 然后根据分析中成功与失败的信息来移动元件参数中心值,使合格率达到最大。若在电路容差域内取 N 个样本点,用蒙特卡罗方法估计电路设计中心为 P^0 时的合格率,则(10.6.3)式可以写为如下形式:

$$Y(P^0,t)=\int_{R_A}\Phi(P,P^0)\,dP=\int_{-\infty}^{\infty}I(P)\,\Phi(P,P^0)\,dP \tag{10.6.5}$$

式中 $I(P)$ 为测试函数, 表示式为

$$I(P)=\begin{cases}1 & \text{当 } P\in R_A\\ 0 & \text{其它}\end{cases}$$

即当取样点在合格域 R_A 之内, 则 $I(P)=1$, 否则为零。此时合格率的估计值 Y 为

$$Y(P^0,t)=\frac{1}{N}\sum_{i=1}^{N}I(p_i) \tag{10.6.6}$$

式中 p_i 为 P 的第 i 个样本点。这就相当于作 N 次实验(计算), 其中成功的次数与总实验次数的比值就是合格率的近似值 $Y(P^0)$。从概率的角度出发, Y 是一个正态分布的随机变量, 其数学期望与方差分别是:

$$E\{Y\}=Y$$

$$D\{Y\}=\sigma^2=\frac{Y(1-Y)}{N} \tag{10.6.7}$$

其中 Y 是实际生产合格率。当 N 越大时, 方差 σ 越小, Y 的值越准确。所以, 为了提高合格率估计值 Y 的置信度, 样本点数 N 必须取得足够大, 因而计算量也很大。这是统计中心设计方法的缺点。近年来, 不少学者对此进行了大量研究, 提出了一系列降低运算量的方法。下面介绍统计中心值优化算法的基本原理和改进措施。

3. 统计搜索法进行最优中心设计

(1) 统计搜索法的基本原理和步骤

这种方法是对电路元件参数取 N 个样本点, 进行 N 次电路分析, 其中合格次数为 N_P, 我们求出合格点在参数空间的重心 G_P; 其中不合格次数为 N_F, 并求出不合格点在参数空间的重心 G_F; 然后沿 G_P 与 G_F 联线方向移动原来的中心值 P^0 的坐标, 使合格率上升。图 10.6.3 是这种算法的示意图。

图 10.6.3 统计搜索法示意图

该算法的具体步骤如下:

1) 给定初始中心值 P^0、参数容差以及电路性能约束条件。并给定允许误差 ε和抽样次数 N。设迭代次数 $k=1$。

2) 用蒙特卡罗法对电路进行 N 次抽样, 并进行电路分析, 得到合格次数 N_P 和不合格次数 N_F。$N_P=\sum_{i=1}^{N}I(p_i)$, $N_F=N-N_P$, $I(p_i)$ 为测试函数。

3) 计算合格点重心 G_P^k 和不合格点重心 G_F^k:

$$g_{Pi}^k=\frac{1}{N_P}\sum_{j=1}^{N_P}I(p_{ij})p_{ij}\qquad i=1,2,\cdots,n \tag{10.6.8}$$

$$g_{Fi}^k=\frac{1}{N_F}\sum_{j=1}^{N_F}(1-I(p_{ij}))p_{ij}\qquad i=1,2,\cdots,n \tag{10.6.9}$$

式中 n 为元件参数数目。则重心为

$$G_P^k = [\,g_{P_1}^k \quad g_{P_2}^k \quad \cdots \quad g_{P_n}^k\,]^T$$

$$G_F^k = [\,g_{F_1}^k \quad g_{F_2}^k \quad \cdots \quad g_{F_n}^k\,]^T \tag{10.6.10}$$

4) 沿 G_F^k 与 G_P^k 联线方向移动 P^0, 移动的迭代公式为:

$$P^{0,k+1} = P^{0,k} + \lambda(G_P^k - G_F^k) \tag{10.6.11}$$

式中 λ 为步长系数, 一般可取 0.1～10 之间。

5) 当满足

$$\|P^{0,k+1} - P^{0,k}\| \leq \varepsilon$$

时迭代结束, $P^{0,k+1}$ 为最优解。否则 $k = k+1$, 返回步骤(2), 继续迭代。

(2) 中心值优化方法的改进

采用以蒙特卡罗分析为基础的中心值优化方法的最大的一个弱点, 就是要在不同的参数中心值上进行抽样点数很大的蒙特卡罗分析, 因而要进行大量的电路模拟, 计算量十分惊人。目前有不少算法对传统的蒙特卡罗方法做了许多改进, 力图降低运算量。我们介绍其中几种。

1) 相关取样方法

在中心值设计过程中, 中心值的移动是 $\Delta Y^{k+1} = Y^{k+1} - Y^k > 0$, 即相邻两次 Y 是增加的。但因为 Y 是个估计值, 当 $\Delta Y > 0$ 时, 有可能实际的 $\Delta Y < 0$, 为了提高算法的置信度, 需提高 ΔY 的精度, 即减小 ΔY 的方差。ΔY 是个正态分布的随机变量, 它的数学期望与方差分别是:

$$E\{\Delta Y^k\} = \Delta Y^k = Y^{k+1} - Y^k \tag{10.6.12}$$

$$D\{\Delta Y^k\} = D\{Y^{k+1}\} + D\{Y^k\} - 2\mathrm{cov}\{Y^{k+1}, Y^k\} \tag{10.6.13}$$

式(10.6.13)中的最后一项为协方差。如果我们在第 k 次取样与第 k+1 次取样时采用“相关取样”, 即相邻两次随机数序列相同, 则协方差一项为正数, 使 ΔY 的方差减小, 从而提高了 ΔY^k 的准确度。在这种情况下, 即使取样点数 N 不很大, 也能保证这种方法的可靠性。故我们可以利用这一特性降低运算量。

2) 中心值移动中利用公共取样点

在中心值设计过程中, 随着中心值的移动, 相邻两次中心值处相应的容差域将有部分重迭, 而且越接近于最优中心值, 重迭的部分也就越大。图 10.6.4 画出了两维空间的情况, 重迭部分为图中的阴影区, 称为公共区。这样, 两次迭代的取样中有一部分样本点将落在公共区内, 这些点称为公用点。若利用这些公用点作为下次迭代中的部分取样点, 显然可以省去不少计算量。而且两次取样存在公共点, 使两次取样存在正相关, 可以提高算法的可靠性, 即可以进一步降低取样点数。

为了有效地利用公用点, 最好取样方式采用均匀抽样。设均匀取样的概率分布密度为 $h(P, P^0)$, 则合格率的估计值

$$Y(P^0) = \frac{1}{N}\sum_{i=1}^{N}\frac{I(p_i)\,\Phi(p_i)}{h(p_i)} \tag{10.6.14}$$

4. 中心值优化方法实例

一个功放 OTL 电路如图 10.6.5 所示, 采用清华大学电子工程系开发的电路模拟程

图 10.6.4　容差域公共区

序 TADS-C4 对此电路进行中心值优化设计。电路中元器件参数的容差为±5%,晶体管的放大倍数的容差是±10%。电路的性能指标约束是:电路直流中点(节点⑥)的电位在 6V±0.1V 范围之内。我们希望通过移动元器件参数的中心值,使电路合格率达到最大。

图 10.6.5　OTL 电路

(1) 先对电路进行蒙特卡罗分析,取样次数为 200,计算出该原始电路的中点电位的中心值及方差,并计算出其生产合格率。

中点电位均值: 6.3433V

中点电位方差: 9.97× 10^{-2}V

合格率: 76%

(2) 对电路进行中心值优化。优化后,元器件参数中心值变化情况如下:

元件参量	原始中心值	优化后中心值
R_1	150kΩ	165.8kΩ
R_2	107kΩ	92.2kΩ
R_3	2.4kΩ	2.46kΩ
R_4	7.5Ω	6.58Ω
R_5	2.7kΩ	2.5kΩ
R_6	1.2kΩ	1.26kΩ
R_7	560Ω	470Ω
R_8	270Ω	221Ω
R_9	15Ω	13Ω
R_{10}	330Ω	288Ω
R_{11}	120Ω	99.5Ω
R_{12}	0.5Ω	0.47Ω
R_{13}	0.5Ω	0.51Ω
β_{T1}	100	107
β_{T2}	60	50
β_{T3}	80	64

(3) 对优化电路进行蒙特卡罗分析。其结果如下。

中点电位均值: 5.9336V

中点电位方差: 8.98×10^{-2}V

合格率: 95.5%　　合格率方差: 1.466%

可见,经过中心值优化电路合格率由76%上升到95.5%。图10.6.6是优化前后电路合格率统计直方图的比较。其中带阴影的直方图是优化后的结果,可以看出其均值在6.0V附近,比优化前结果改善很多。

图10.6.6　优化前后统计直方图

10.7 模拟退火法

模拟退火法(simulated annealing method)是一种全局优化算法,它比较适合于求解较大型的优化问题,特别是组合优化问题。近年来这种算法在电路优化问题中得到相当的重视和应用。

1. 算法原理

模拟退火法的思想来源于金属冷却退火的自然现象,金属在高温下熔化,由于金属分子具有很大的平均动能,每个分子几乎可以随处移动,如果让金属非常缓慢地冷却,则熔融金属分子将规则地排列起来形成单晶体。这种缓慢冷却的过程称为退火。从能量的角度而言,系统由高能状态经缓慢冷却后达到其最低的能量状态。当然,如果是快速冷却,则熔融金属将成为多晶体,甚至非晶体,处于这种状态的系统能量将比单晶态时要高一些。

将上述退火过程中的系统能量与优化问题的目标函数联系起来,将退火过程中分子的移动与优化设计参数的变化联系起来,将退火过程中温度的变化速度与优化迭代过程中的步长联系起来,让优化过程模拟退火过程,即让迭代步长缓慢变化,并且设计参数以某种概率随机移动(就像金属分子无规则热运动一样),那么,就有可能获得一个最小的目标函数值,达到该优化问题的全局最优解。这就是模拟退火法的基本思想。

当温度为 T 时,由波尔兹曼分布,一个系统的能量 E 的概率与 $e^{-E/kT}$ 成正比,其中 k 为波尔兹曼常数。因此,系统能量由 E_1 状态变化到 E_2 状态的概率为:

$$P = e^{-(E_2-E_1)/kT} \tag{10.7.1}$$

当 $E_2 < E_1$ 时,$P > 1$,这表明由能量高的状态变化到能量低的状态总是允许的,而且我们的目的就是要达到能量最低状态。当 $E_2 \geq E_1$ 时,P 比较小,但不为零;也就是说退火过程中系统能量稍有升高也是允许的。这一准则在优化过程中体现为:目标函数呈下降趋势,但也允许目标函数低的优化参数向目标函数高的优化参数移动。这是模拟退火法区别前面优化算法的关键所在。也正因为此,模拟退火法有可能跳出某个局部极小点,达到另一个更好的局部极小点,甚至是全局极小点。

我们采用模拟退火法解决连续函数的极值问题。假定在 N 维空间中,目标函数 $f(x)$ 有多个极值点,设计变量 x 代表系统状态,$f(x)$ 代表系统能量 E,控制参数为 T,$f(x)$ 按退火策略缓慢减小。为了模拟分子的热运动,需要随机地产生 x 的移动量 Δx,并根据 $f(x+\Delta x)-f(x)$ 来确定设计变量 x 移动到 $x+\Delta x$ 的这一个随机事件在当前实验中是否发生,从而模拟一个退火过程。

我们以 10.4 节介绍的单纯形法为基础,首先由 $(N-1)$ 个初始点构造一个单纯形,通过反射、延伸、压缩等措施获取一个更好的单纯形。在更新单纯形时,不仅仅是用使目标函数下降的点去替换原单纯形中目标函数最大的那个顶点,而且是按一定的概率接收一个或一系列新的顶点,即使是目标函数值较大的点也有机会被用来更新原单纯形的顶点,这个过程就像单纯形在做布朗运动一样。因此,在温度条件控制之下,反复进行足够多次的布朗运动,就能很好地模拟退火过程,而最终获得的单纯形将收敛于 $f(x)$ 的全局极小值点。

2. 退火策略

采用模拟退火法求解连续函数的极值问题,其退火策略是多种多样的。下面我们列举三种常用的策略:

(1) 单纯形每移动 m 次,将温度 T 降到(1- ε)T。其中,ε,m 都可以根据统计方法估计出一个适当的取值。

(2) 假定单纯形总共移动 M 次之后就停止优化过程,而每移动 m 次,温度 T 就降到

$$T = T_0(1 - k/M)^{\alpha} \tag{10.7.2}$$

其中,k 是到当前为止进行随机移动的次数,它应是 m 的倍数。α是与目标函数极值点分布有关的常数,通常取 α= 1, 2, 或 4 等值。由(10.7.2)式可以看出,当给定 M 之后,α越大则在低温下随机移动的次数也就越多。因为,温度 T 的下降量随 α增加而增加。

(3) 单纯形每移动 m 次,将温度 T 降到

$$T = \beta(f_a - f_b) \tag{10.7.3}$$

其中,β是经验常数,f_a 是当前单纯形各顶点的目标函数的最小值,f_b 则是到目前为止所获得的目标函数的最小值。

3. 收敛策略

采用模拟退火法进行优化迭代,其收敛判据也是多种多样的,下面是几种常用的收敛策略。

(1) 假定单纯形各顶点处的最大目标函数为 f_{hi},最小目标函数为 f_{lo},若

$$\frac{2\times |f_{hi} - f_{lo}|}{|f_{hi}| + |f_{lo}|} \leq \varepsilon \tag{10.7.4}$$

成立,则可以停止优化迭代过程。式中,ε为给定的误差精度要求。

(2) 给定一个单纯形总共移动的最大次数限制 M,移动次数达到 M 后,优化过程就停止。

(3) 给定一个最低温度限制 T_{min},当温度逐渐降低到 T_{min} 后,优化过程停止。

(4) 假定在单纯形移动 m 次过程中,式(10.7.4)所反映的误差始终保持在一个量级上,不再改善,则可以考虑中止优化过程。

上面几种收敛策略也可以结合起来应用,以提高求解效率。

4. 模拟退火法的算法步骤

由前面介绍的模拟退火法的原理和优化策略,可将以单纯形法为基础的模拟退火算法步骤归纳如下:

(1) 对目标函数 f(X),构造一个初始单纯形,它由(N+ 1)个顶点描述,N 是 X 的维数。并给定温度初值 T= T_0。

(2) 计算单纯形各顶点的目标函数值,找出单纯形各顶点处最大目标函数 f_{hi}和最小目标函数 f_{lo},若满足

$$\frac{2\times |f_{hi} - f_{lo}|}{|f_{hi}| + |f_{lo}|} \leq \varepsilon$$

则停止优化。

(3) 单纯形作 m 次移动,根据概率确定新的单纯形顶点。

1) 随机地产生移动向量 ΔX, 并按

$$P = \exp[-(f(X+\Delta X)-f(X))/(kT)]$$

计算概率 P。

2) 产生随机数 α $\alpha\in[0,1)$。若 $\alpha>P$, 则接收 $(X+\Delta X)$, 产生一个新的单纯形顶点; 否则拒绝接收。

3) 重复上面 1), 2) 步骤共 M 次。

(4) 按照退火策略, 计算控制温度 T, 例如用公式(10.7.3)计算:

$$T = \beta(f_a - f_b)$$

(5) 返回步骤(2)。

习　　题

10.1　用黄金分割法求函数 $f(x)=(x-4)^2$ 的极小点。设区间长度为[0, 10], 要求搜索三次。

10.2　用黄金分割法求函数 $f(x)=x(x+2)$ 在区间[-3, 5]内的极小点, 要求极小点所在区间长度小于 0.05。

10.3　用二次插值算法求函数 $f(x)=\frac{1}{3}x^3-x^2+1$ 在区间[0, 3]内的极小点。要求误差 $\varepsilon=0.05$, 初始值 $x^0=1$。

10.4　用最速下降法求下列函数的极小点:

(1) $f(X)=x_1^2+25x_2^2$

设初值 $X^0=[2,2]^T$, 迭代二次。

(2) $f(X)=4x_1^2+x_2^2-x_1x_2$

设初值 $X^0=[1,1]^T$, 迭代二次。

(3) $f(X)=(x_1-1)^2+(x_2-2)^2$

设 $X^0=[0,0]^T$, 迭代二次。

10.5　用牛顿法求函数 $f(X)=(x_1-2)^4+(x_1-2x_2)^2$ 的极小点, 设初值 $X^0=[0,3]^T$, 要求误差 $\varepsilon=0.08$。

10.6　求函数 $f(X)=x_1^2+2x_1x_2+2x_2^2=\frac{1}{2}X^THX$ 的极小点。设 $X^0=[1,-2]^T$, $H=\begin{matrix}2 & 2\\ 2 & 4\end{matrix}$。

(1) 用最速下降法, 迭代二次;

(2) 用牛顿法;

(3) 用共轭梯度法。

参 考 文 献

[1] Hachtel G D, Brayton R K, Gustavson F G. The Sparse Tableau Approach to Network Analysis and Design. IEEE Trans. on Circuit Theory, 1971, CT-18: 101—113

[2] Ho C W, Ruehli A E, Brennan P A. The Modified Nodal Approach to Network Analysis. IEEE Trans. on Circuits and Systems, 1975, CAS-22: 504—509

[3] Chua L O, Lin P M. Computer-Aided Analysis of Electronic Circuits: Algorithms and Computation Techniques. Prentice-Hall, 1975

[4] Mocalla W J, Pedreson D O. Elements of Computer-Aided Circuit Analysis, IEEE Trans. on Circuit Theory, 1971, CT-18(1)

[5] Nagel L W. SPICE2: A Computer Program to Simulate Semiconductor Circuits. Memorandum No. ERL. M520, UC Berkeley, 1975

[6] Berry R D. An Optimal Ordering of Electronic Circuit Equations for a Sparse Matrix Solution. IEEE Trans. on Circuit Theory, 1971, CT-18: 40—45

[7] Nakhla M, Singhal K, Vlach J. An Optimal Pivoting Order for the Solution of Sparse Systems of Equations. IEEE Trans. on Circuits and Systems, 1974, CAS 21: 222—225

[8] Forsythe G E, Malcolm M A, Moler C B. Computer Methods for Mathematical Computations. Prentice-Hall, 1977

[9] Vlach J, Sighal K. Computer Methods for Circuit Analysis and Design. Van Nostrand Reinhold Company, 1983

[10] 洪先龙等. 计算机辅助电路分析. 清华大学出版社, 1983

[11] Antognetti P, Massobrio G. Semiconductor Device Modeling with SPICE, McGraw-Hill Book Company, 1988

[12] Connelly J A, Choi P. Macromodeling with SPICE, Prentice-Hall, 1992

[13] Getreu L E. CAD of Electronic Circuits, 1 Modeling the Bipolar Transistor. Elsevier Scientific Publishing Company, 1978
周宁华, 陈幼松译. 双极型晶体管模型. 科学出版社, 1981

[14] Poon H C. Modeling of Bipolar Transistor using Integral Charge-Control Model with Application to Third-Order Distortion Studies, IEEE Trans. on Electron Devices, 1972, ED-19: 719—731

[15] Merckel G, Borel J, Cupcea H Z. An Accurate Large-Signal MOS Transistor Model for Use in Computer Aided Design. IEEE Trans. on Electron Devices, 1972, ED-19: 681—690

[16] Sinencio E S, Majewski M L. A Nonlinear Macromodel of Operational Amplifiers in the Frequency Domain. IEEE Trans. on Circuits and Systems, 1979, CAS-26: 395—402

[17] Chien M J, Kuh E S. Solving Nonlinear Resistive Network Using Piecewise-linear Analysis and Simplical Subdivision. IEEE Trans. on Circuits and Systems, 1977, CAS-24: 305—317

[18] 董在望, 尹达衡. 模拟集成电路原理与系统. 高等教育出版社, 1987

[19] 夏武颖. 半导体器件模型和工艺模型. 科学出版社, 1986

[20] Matsuno C T, Sharma A K, Oki A K. A Large Signal HSPICE Model for the Heterojunction Bipolar Transistor. IEEE Trans. Microwave & Techniques, 1989, 37(9): 1472—1475

[21] Zaghloul M E, Brgant P R. Error Bounds on Solutions of Nonlinear Networks When Using Approximate Element Characteristics. IEEE Trans. on Circuits and Systems, 1980, CAS-27: 20—29

[22] Brayton R K, Gustavson F G, Hachtel G D. A New Efficient Algorithm for Solving Differential-Algebraic Systems Using Implicit Backward Differentiation Formulas. Proc. IEEE, 1972, 60(1): 98—108

[23] Brayton R K et al. The Methods for Variable-Order Variable-Step Backward Differentiation Methods for Nonlinear Electrical Network. Internat IEEE Conf. Systems, Networks and Computers, 1971

[24] Stoer J, Bulirsch R. Introduction to Numerical Analysis, Chapter 7. Springer-Verlag, 1980

[25] Kahaner D, Moler C, Nash S. Numerical Methods and Software, Chapter 8. Prentice Hall, 1989

[26] Fidler J K. Network Sensitivity Calculation. IEEE Trans. on Circuits and Systems, 1976, CAS-23: 567

[27] Brayton R K, Spence R. Sensitivity and Optimization. EL seriver, 1980

[28] Vlach j, Singhal K. Sensitivity Minimization of Networks with Operational Amplifiers and Parasitics, IEEE Trans. on Circuits and Systmes, 1980, CAS-27 (8): 688

[29] Graut L G, Sewell J I. A Theory of Equivalent Active Networks. IEEE Trans. on Circuits and Systems, 1976, CAS-23(6): 350

[30] Cheetham B M G. A New Theory of Continously Equivalent Networks. IEEE Trans. on Circuits and Systems, 1974, CAS-21(1): 17

[31] 徐钟济. 蒙特卡罗方法. 上海科学技术出版社, 1985

[32] 程兴新, 曹敏. 统计计算方法. 北京大学出版社, 1989

[33] 浙江大学数学系高等数学教研组. 概率论与数理统计. 人民教育出版社, 1979

[34] 凌燮亭. 电路参数的容差分析与设计. 复旦大学出版社, 1991

[35] Kolev L V, Mladenov V M, Vladov S S. Interval Mathematics Algorithms for Tolerance Analysis. IEEE Trans. on Circuits and Systems, 1988, CAS-35(8): 967

[36] Schjaer-Jacobsen H, Madsen K. Algorithms for Worst-Case Tolerance Optimization. IEEE Trans. on Circuits and Systems, 1979, CAS-26: 775

[37] Singhal K, Pipel J F. Statistical Design Centering and Tolerancing Using Parametric Sampling. IEEE Trans. on Circuits and Systems 1981, CAS-28: 692

[38] Ling X T. Worst-Case Tolerance Design by the Interval Algebra. Proc. IEEE Int. Symp. Circuit and Systems, 1981: 217

[39] **方再根. 计算机模拟和蒙特卡罗方法. 北京工业学院出版社**, 1988

[40] Rabbat N B, Sangiovanni-Vincentlli A L, Hsieh H Y. A Multilevel Newton Algorithm with Macromodeling and Latency for the Analysis of Large-Scale Nonlinear Circuits in the Time Domain. IEEE Trans. on Circuits and Systems, 1979, CAS-26: 733—741

[41] Yang P, Hajj I N, Trick T N. SLATE: A Circuit Simulation Program with Latency Exploitation and Nodal Tearing. Proc. IEEE Int. Conf. Circuits and Computers, 1980: 353—355

[42] Wu F F. Solution of Large-Scale Networks by Tearing, IEEE Trans. on Circuits and Systems, 1976, CAS-23: 706—713

[43] Newton A R, Sangiovanni-Vincentelli A L. Relaxation-Based Electrical Simulation. IEEE Trans. CAD, 1984, CAD-3: 308—331

[44] Lelarasmee E, Ruehli A E, Sangiovanni-Vincentelli A L. The Waveform Relaxation Method for Time-Domain Analysis of Large Scale Integrated Circuits, IEEE Trans. CAD, 1982, CAD-1: 131—145

[45] White J, Sangiovanni-Vincentelli A L. RELAX: A New Circuit Simulator for Large-Scale MOS Integrated Circuits. Proc. 19th Design Automation Conf. 1982: 682—690

[46] **彭沛. 大规模电路的电路模拟技术. 东南大学出版社**, 1989

[47] **陈宝林. 最优化理论与算法. 清华大学出版社**, 1989

[48] **蔡宣三. 最优化与最优控制. 清华大学出版社**, 1982

[49] Kirkpatrick S, Gelatt C D, Vecchi M P. Optimization by Simulated Annealing. Science, 1983 220: 671—680

[50] Press W H et. al. Numerical Recipes In C. 2nd ed. Cambridge Univ. Press, 1992

[51] **高葆新等. 微波电路计算机辅助设计. 清华大学出版社**, 1988

附录　教学软件

```
/*          Voltage source: --- Vxxxx node1 node2 value;
            Risistor      : --- Rxxxx node1 node2 value;
            Current source: --- Ixxxx node1(+) node2(-) value;
            Inductor      : --- Lxxxx node1 node2;
            Capacity      : --- Cxxxx node1 node2;
            VCCS          : --- Gxxxx nodek nodel nodei nodej value(g);
            VCVS          : --- Exxxx nodek nodel nodei nodej value(u);
            CCCS          : --- Fxxxx nodek nodel nodei nodej value(b);
            CCVS          : --- Hxxxx nodek nodel nodei nodej value(r);
            Diode         : --- Dxxxx node1(+) node2(-) [Rs] [Is]
                                    (default: Rs= 1, Is= 1e- 14); */
# include "cad.h"
void Initial(void);
void fill_dio(void);
/* -------------------------main program----------------------* /

void main(int argc, char * argv[])
{     int m, i, true= 0;
        if (argc= = 2)
                        {
                        FP_in= fopen(argv[1], "r");
                        if (FP_in= = NULL) {printf("Cannot open IN file\n"); exit(0);}
                        }
        else          {printf("Please input data by screen \n");
                       FP_in= stdin;                    }
        CAD_in();               /* get data* /
        do
       {
        Initial();              /* initiate the A and B matrices* /
        CAD_fill();             /* fill in linear device data* /
        fill_dio();             /* fill in diode data* /
        CAD_lu();
        true= CONVERGE();       /* results converge or not? * /
        if(NUM_dio= = 0)true= 0;  /* if no diode, certainly converge, then exit* /
       }
        while(true);
CAD_out_1();                    /* output citcuit data information* /
CAD_out_2();                    /* output result* /
}
/* --------------main program end-------------* /
```

```
/* ------------initial A and B matrices------ */
void Initial(void)
{int i, j;
for(i= 0;i< 30;i+ + )
  {for(j= 0;j< 30;j+ + )
          A[i][j] = 0.0;
  B[i]= 0.0;
  }
NUM_new= 0;
}

/* --------fill all diode-------* /
void fill_dio(void)
{int m= 0;
              while(m< NUM_dio)
                    { Fill_diode(m); m+ + ;}
}

/* ------get data from screen or input file--------* /
void CAD_in()
{
          int       i= 0, j= 0, l= 0, flag= 0;
          char      NAME[5];
          float     var;
  while(flag= = 0)
  {       fscanf(FP_in, "%s",NAME);
                    NUM_unit+ + ;
  switch(NAME[0])
   {
                    default:
                                        {printf( "\n There is no such component\n");        break;}
        case 'Z' :case 'z' :
                                        {flag= 1;NUM_unit- -    ;                   break;}
        case 'L' :case 'l' :
        case 'C' :case 'c' :
                                        {fscanf(FP_in, "%d %d", &L[i].node_i, &L[i].node_j);
                                          strcpy(L[i].name,NAME);
                                          i+ + ;break;
                                        }
        case 'R' :case 'r' :
        case 'V' :case 'v' :
        case 'I' :case 'i' :
                                        {fscanf(FP_in, "%d %d %g",      &L[i].node_i,      &L[i].
        node_j,      &var);
                                           strcpy(L[i].name,NAME);
                                           L[i].value= var;
                                           i+ + ;
                                           break;
                                        }
```

```
  case 'G': case 'g':

  {fscanf(FP_in, "%d%d%d%d%g",&CS[j].node_k, &CS[j].node_l, &CS[j].node_i, &CS[j].node_
  j, &var);
                                        strcpy(CS[j].name, NAME);
                                        CS[j].value= var;
                                        j++;break;
                                    }
  case('D'):case('d'):
            {DIO[l].is= 1e-14;DIO[l].rs= 1.0;DIO[l].vd= 0.0;/* default value* /
            fscanf(FP_in, "%d%d%f%g%f",&DIO[l].node_i, &DIO[l].node_j, &DIO[l].rs, &DIO
                  [l].is, &DIO[l].vd);
            strcpy(DIO[l].name, NAME);
            if(DIO [l].is> 1e-9)
                  DIO[l].is= 1e-14;
            l++;break;}
  }
}
NUM_linear= i; NUM_cs= j; NUM_dio= l;/* store every type device's number* /
/* every node number - 1, so they can be easily dealed with* /
for(i= 0;i< NUM_dio;i++)
      {DIO[i].node_i--;DIO[i].node_j--;}
for(i= 0;i< NUM_linear;i++)
{    L[i].node_i--;      L[i].node_j--; }
for(i= 0;i< NUM_cs;i++)
{  CS[i].node_i--;          CS[i].node_j--;       CS[i].node_k--;       CS[i].node_l
--;}
}
/* ----------------fill linear device----------------* /
void CAD_fill()
{
      int m, i;
      NUM_node= MAX_node();              /* calculate total nodes number* /
      NUM_dim= NUM_node;       /* so decide dimension of A and B matrix* /
      for(m= 0;m< NUM_linear;m++)
      {switch(L[m].name[0])
            {
            case 'R': case 'r':
            case 'L': case 'l':
            case 'C': case 'c':
                                    {Fill_r(m); break;}
            case 'I': case 'i':
                                    {Fill_i(m); break;}
            case 'V': case 'v':
                                    {Fill_v(m); break;}
            }
      }
      for(m= 0;m< NUM_cs;m++)
            Fill_vccs(m);
```

```
}
/* ----------to find out the largest node number ----------------* /
MAX_node()
{       int m,n= 0;
        for (m= 0;m< NUM_linear;m+ + )
                {
                if (n< L[m].node_i) n= L[m].node_i;
                if (n< L[m].node_j) n= L[m].node_j;
                }
        for (m= 0;m< NUM_cs;m+ + )
                {
                if (n< CS[m].node_i)        n= CS[m].node_i;
                if (n< CS[m].node_j)        n= CS[m].node_j;
                if (n< CS[m].node_k)        n= CS[m].node_k;
                if (n< CS[m].node_l)        n= CS[m].node_l;
                }
        for (m= 0;m< NUM_dio;m+ + )
                {
                if (n< DIO[m].node_i) n= DIO[m].node_i;
                if (n< DIO[m].node_j) n= DIO[m].node_j;
                }
        return n;
}
/* ----------------fill Resistor ----------------* /
void Fill_r(int m)
{
        int x,   y;                float var;
        if (L[m].name[0]= = 'L' ||L[m].name[0]= = 'l') L[m].value= R_l;
        if (L[m].name[0]= = 'C' ||L[m].name[0]= = 'c') L[m].value= R_c;
        x= L[m].node_i; y= L[m].node_j;
        if (L[m].value> = 1.e- 9) var= 1/L[m].value;
        else
                {printf("Component %d is shot-cut\n",m);
                 exit(0);}
        if (x! = - 1)
                {
                        A[x][x]= A[x][x]+ var;
                        if        (y! = - 1)
                        {         A[y][x]= A[y][x]- var;
                                  A[x][y]= A[x][y]- var;
                                  A[y][y]= A[y][y]+ var;
                        }
                }
        else
                {       if (y! = - 1)       A[y][y]= A[y][y]+ var;     }
}
/* ------------fill in current source data----------------------* /
void Fill_i(int m)
{       if       (L[m].node_i! = - 1)       B[L[m].node_i]= - L[m].value;
```

```
        if (L[m].node_j != -1) B[L[m].node_j] = +L[m].value;
}
/* ----------------------fill v ----------------------* /
void Fill_v(int m)
{       int     x,y,z,flag;
        x= L[m].node_i;         y= L[m].node_j; flag= 0;
/* ------if at the same two nodes there is already a voltage source and the new added voltage source's
        value is not as the same as the old one, then the circuit data has error. Because the same two
        nodes cannot has two diffrent voltage---------* /
        for (z= 0;z<=NUM_new-1;z++)
  {
    if (I[z].node_i==x&&I[z].node_j==y||I[z].node_i==y&&I[z].node_j==x)
            {
                if ( ABS(B[I[z].line_num]-L[m].value)>1.e-3 )
                    {printf("Error in circuit design on V\n");
                    exit(0);}
                else
                    {flag=1;            break;}
            }
  }
  if (flag==0)
  {
        NUM_dim++;      /* add a new line in matrix* /
        I[NUM_new].line_num=NUM_dim;
        I[NUM_new].node_i=x;            I[NUM_new].node_j=y;
        NUM_new++;
        B[NUM_dim]=B[NUM_dim]+L[m].value;
        if      (x!=-1)     {
                                A[x][NUM_dim]=A[x][NUM_dim]+1;
                                A[NUM_dim][x]=A[NUM_dim][x]+1;
                                };
        if      (y!=-1)     {
                                A[y][NUM_dim]=A[y][NUM_dim]-1;
                                A[NUM_dim][y]=A[NUM_dim][y]-1;
                                }
  }
}

/* --------fill vccs---------* /
void Fill_vccs(int m)
{
        int     i,j,k,l;                float   var;
        i=CS[m].node_i; j=CS[m].node_j; k=CS[m].node_k; l=CS[m].node_l;
        var=CS[m].value;
        if      (i!=-1&&k!=-1) A[k][i]=A[k][i]+var;
        if      (i!=-1&&l!=-1) A[l][i]=A[l][i]-var;
        if      (j!=-1&&k!=-1) A[k][j]=A[k][j]-var;
        if      (j!=-1&&l!=-1) A[l][j]=A[l][j]+var;
}
```

```
/* -------fill diode------* /
void Fill- diode(l)
{
float rs, dis, gd, id, vd; int y, i1, j1;
CAD- DIO(l); /* before fill in, goto CAD- DIO() to deal with the data* /
NUM- dim+ + ; /* add a new line to matrix, because of the diode's RS* /
DIO[l]. line- num= NUM- dim;
y= DIO[l]. line- num;
gd= DIO[l]. gd;
id= DIO[l]. id;
rs= DIO[l]. rs;
i1= DIO[l]. node- i;
j1= DIO[l]. node- j;
if (i1! = - 1)
  {A[i1][i1]= A[i1][i1]+ 1/rs;
  A[i1][y]- = 1/rs;
  A[y][i1]- = 1/rs;
  }
A[y][y]+ = 1/rs+ gd;
if (j1! = - 1)
  {A[y][j1]- = gd;
  A[j1][y]- = gd;
  A[j1][j1]+ = gd;
  B[j1]+ = id;}
B[y]- = id;
}
/* ---to deal with the diode data---* /
void CAD- DIO(int l)
{float diq= 1e- 5;
float dis, vq, vd= 0. 0;
double long id, dj, gd;
int i, j;
dis= DIO[l]. is;
vq= log(1+ diq/dis)* vt;
i= DIO[l]. line- num;
vd= LB[i];
j= DIO[l]. node- j;
if(j! = - 1)vd-= LB[j];
if(iter= = 0)vd= DIO[l]. vd;
if(iter= = 0&&vd= = 0. 00)vd= vq;
if(iter! = 0 )
            {id= DIO[l]. gd* vd+ DIO[l]. id;
            if( vd> vq&&id> = diq)vd= vt* log( id/dis+ 1);
            else
             if(vd> vq)vd= vq;
             }
gd= 0. 0;
if(vd/vt> - 100)gd= dis/vt* exp(vd/vt);
if(vd/vt> - 100&&gd< 1e- 14)gd= 1e- 14;
```

```
dj= gd* (vt- vd)- dis;
DIO[l].gd= gd;
DIO[l].id= dj;
DIO[l].vd= vd;
}
/* -------- CAD- lu, use LU methods to solve formula matrices A and B-------* /
void CAD- lu()
{       int       Line- num[10], Col- num[10];
        int       MAX- line, MAX- col, NR;
        int       i, j, k, x;
        float     var, sum, MAX- a;
        NR= NUM- dim;
        for(i= 0; i< = NR; i+ + )
                {       Line- num[i]= i; Col- num[i]= i;   }
        for(k= 0; k< = NR- 1; k+ + )
        {
                MAX- a= A[k][k]; MAX- line= k; MAX- col= k;
                for(i= k; i< = NR; i+ + )
                        {for(j= k; j< = NR; j+ + )
                                {var= (float)fabs( (double)A[i][j] );
                                if (var> MAX- a)
                                        {
                                        MAX- a= var;
                                        MAX- line= i;      MAX- col= j;
                                        }
                                }
                        }
                for(j= 0; j< = NR; j+ + )
                        {var= A[k][j];          A[k][j]= A[MAX- line][j];
        A[MAX- line][j]= var;}
                        x= Line- num[k]; Line- num[k]= Line- num[MAX- line];
        Line- num[MAX- line]= x;
                for(i= 0; i< = NR; i+ + )
                        {var= A[i][k];          A[i][k]= A[i][MAX- col];
        A[i][MAX- col]= var;}
                        x= Col- num[k];          Col- num[k]= Col- num[MAX- col];
        Col- num[MAX- col]= x;
                for(i= k+ 1; i< = NR; i+ + )
                        A[i][k]= A[i][k]/A[k][k];
                for(i= k+ 1; i< = NR; i+ + )
                        {for (j= k+ 1; j< = NR; j+ + )
                            A[i][j]= A[i][j]- A[i][k]* A[k][j];
                            }
        }
        for(i= 0; i< = NR- 1; i+ + )
                {
                x= Line- num[i]; var= B[i];
                B[i]= B[x];                B[x]= var;
                for(j= i+ 1; j< = NR; j+ + )
```

```
                    if (Line_num[j]==i)        Line_num[j]=x;
            }
        for(i=1;i<=NR;i++)
            {
            sum=0;
            for(j=0;j<=i-1;j++)
                sum=sum+A[i][j]*B[j];
            B[i]=B[i]-sum;
            }
        if (A[NR][NR]<1.E-6&&A[NR][NR]>-1.E-6)
            {
                printf("Error in result: no exact answer\n");
                exit(0);
            }
        B[NR]=B[NR]/A[NR][NR];
        for(i=1;i<=NR;i++)
            {
            sum=0;
            for(j=NR-i+1;j<=NR;j++)
                sum=sum+A[NR-i][j]*B[j];
            if (A[NR-i][NR-i]<1E-6&&A[NR-i][NR-i]>-1.e-3)
                {
                printf("Error in result: no exact answer\n");
                exit(0);
                }
            B[NR-i]=(B[NR-i]-sum)/A[NR-i][NR-i];
            }
        for(i=0;i<=NR;i++)
            Line_num[i]=Col_num[i];
        for(i=0;i<=NR;i++)
            {for(j=i;j<=NR;j++)
                {if (Line_num[j]==i)
                    {
                    var=B[i];       B[i]=B[j];
                    B[j]=var;       Line_num[j]=Line_num[i];
                    }
                }
            }

}
/* ------judge the results converge or not---*/
int CONVERGE()
{int i;
float vk;
float ebsr=0.0;
for(i=0;i<30;i++)
   {vk=fabsl(LB[i]-B[i]);
  if(ebsr<vk)
     ebsr=vk;
```

```
   LB[i]= B[i];       /* keep the previous results* /
    }
iter+ + ;
if (ebsr< = err)
   {printf( "result: NR total iter= \t %d\n",iter);
   return 0;
   }
if (ebsr> err)
   {if(iter> itermax)
      {printf( "Iter> %d, not converge\n",itermax);
      return 0; }
      }
return 1;
}
/* -------- output circuit data information----------* /
void CAD- out- 1()
{
        int i, j;
        printf( "The total number of component is: %d\n",NUM- unit);
        printf( "the number of line2 is: %d\n",NUM- linear);
        for (i= 0;i< NUM- linear;i+ + )
                printf( "%s\t%d\t%d\t%f\n",L[i]. name, L[i]. node- i+ 1, L[i]. node- j+ 1, L[i].
        value);
        printf( "the number of cs      is: %d\n",NUM- cs);
        for (i= 0;i< NUM- cs;i+ + )

        printf( "%s\t%d\t%d\t%d\t%d\t%f\n",CS[i]. name, CS[i]. node- i+ 1, CS[i]. node- j+ 1,
        CS[i]. node- k+ 1, CS[i]. node- l+ 1, CS[i]. value);
        for (i= 0;i< NUM- dio;i+ + )

        printf( "%s\t%d\t%d\t%f\t%g\n",DIO[i]. name, DIO[i]. node- i+ 1, DIO[i]. node- j+ 1,
        DIO[i]. rs, DIO[i]. is);
}
/* -----output result----* /
void CAD- out- 2()
{       int i;
        for (i= 0;i< = NUM- node;i+ + )
                printf( "V%d= %g\n",i+ 1,B[i]);
        for (i= 0;i< NUM- new;i+ + )
                {
                printf( "I(%d,%d)= %g\n",I[i]. node- i+ 1, I[i]. node- j+ 1, B[NUM- node+ 1
+ i]);
                }
}

/* ----------end of program----------* /
# include < stdio. h>

# include < string. h>
```

```
# include < math.h>
# include < stdlib.h>
# define        R_l       1.e- 1 /* resistor value of inductor* /
# define        R_c       1.e+ 7 /* resistor value of capacitor* /
# define ABS(x)((x)> = 0? (x): -(x))
void   CAD_in (void);          /* function to get circuit data* /
void   CAD_fill(void);         /* function to fill in the formula matrix* /
void   CAD_lu (void);          /* function to solve formula ---LU method* /
void   CAD_out_1(void);        /* output the circuit imformation* /
void   CAD_out_2(void);        /* output the circuit analysis result* /
void   Fill_r(int);            /* fill in the resistor data to matrix * /
void   Fill_v(int);            /* fill in the voltage source data to matrix* /
void   Fill_i(int);            /* fill in the current source data to matrix* /
void   Fill_vccs(int);         /* fill in the voltage-controlled current source* /
void   CAD_DIO(int);           /* deal with the diode data before fill in* /
int    CONVERGE(void);         /* function to find if the result converge* /
void   Fill_diode(int);        /* fill in the diode data to matrix* /
int    MAX_node(void);         /* find out the largest node number* /
static int
       NUM_node,               /* total nodes * /
       NUM_dim,                /* dimension of the matrix A and B* /
       NUM_unit,               /* number of elements* /
       NUM_linear,             /* number of linear device : V, I, R, L, C* /
       NUM_new,                /* number of new fill-in lines* /
       NUM_cs,                 /* number of VCCS* /
       NUM_dio;                /* number of diode* /
struct Line2                   /* struction of two-node linear device* /
       {
       char                    name[5];
       int                     node_i;
       int                     node_j;
       float                   value;
       }L[10];
struct Line4                   /* struction of four-node linear device* /
       {
       char                    name[5];
       int                     node_i;
       int                     node_j;
       int                     node_k;
       int                     node_l;
       float                   value;
       }CS[10];
struct Diode                   /* struction of diode* /
     {
       char name[5];
       int node_i;
       int node_j;
       float rs;
       double is;
```

```
        float vd;
        double long id;
        double long gd;
        int line_num;
        }DIO[10];
struct I_new_fill                    /* struction of new-added line* /
        {
        int                          line_num;
        int                                node_i;
        int                                node_j;
        }I[10];
float
        A[30][30], B[30],       /* matrix of A and B---circuit formula* /
        LB[30];                 /*  vb[30] is used to keep the previous B[30]* /
int iter= 0, itermax= 150;
float vt= 0.026, err= 0.0001;
FILE * FP_in;
```