

信息技术和电气工程学科国际知名教材中译本系列

哈尔滨工程大学研究生规划教材

Understanding the Finite-Difference Time-Domain Method

时域有限差分方法及编程技巧

[美] John B. Schneider 著

余文华 李文兴 张朝柱 译

刘培国 校

清华大学出版社



信息技术和电气工程学科国际知名教材

Understanding the Finite-Difference Time-Domain Method
时域有限差分方法及编程技巧



[美] John B. Schneider 著
余文华 李文兴 张朝柱 译

清华大学出版社
北 京

内 容 简 介

《时域有限差分方法及编程技巧》是一部详细介绍时域有限差分方法、编程技巧和应用的著作。该书深入浅出地介绍了时域有限差分基本理论、关键技术、编程技巧以及工程应用。该书配有一维、二维和三维时域有限差分原程序并详细地解释了编程方法和技巧。

该书非常适合研究生和年轻教师用作计算电磁学教材以及科学研究或者工程应用的参考书。

北京市版权局著作权合同登记号 图字：01-2015-0655

本书封面贴有清华大学出版社防伪标签，无标签者不得销售。

版权所有，侵权必究。侵权举报电话：010-62782989 13701121933

图书在版编目(CIP)数据

时域有限差分方法及编程技巧/(美)施奈德 (Schneider, J. B.) 著；余文华，李文兴，张朝柱译.--北京：清华大学出版社，2015

信息技术和电气工程学科国际知名教材中译本系列

ISBN 978-7-302-37771-9

I. ①时… II. ①施… ②余… ③李… ④张… III. 有限差分 IV. ①O175.7

中国版本图书馆 CIP 数据核字(2014)第 190523 号

责任编辑：文 怡

封面设计：张海玉

责任校对：李建庄

责任印制：杨 艳

出版发行：清华大学出版社

网 址：<http://www.tup.com.cn>, <http://www.wqbook.com>

地 址：北京清华大学学研大厦 A 座

邮 编：100084

社 总 机：010-62770175

邮 购：010-62786544

投稿与读者服务：010-62776969, c-service@tup.tsinghua.edu.cn

质量反馈：010-62772015, zhiliang@tup.tsinghua.edu.cn

印 刷 者：三河市君旺印务有限公司

装 订 者：三河市新茂装订有限公司

经 销：全国新华书店

开 本：185mm×260mm

印 张：20.75

字 数：513 千字

版 次：2015 年 3 月第 1 版

印 次：2015 年 3 月第 1 次印刷

印 数：1~2000

定 价：49.00 元

产品编号：058566-01

译 者 序

华盛顿州立大学 John Schneider 教授的《Understanding Finite Difference Time Domain Method》是一本非常难得的计算电磁学入门的优秀教材。该书除了由简到繁系统地介绍时域有限差分理论和方法外,从软件工程的角结合理论和方法,指导学生一步一步编写一个优秀的时域有限差分应用程序,使学生得到了时域有限差分方法和软件工程两方面的学习和训练。该书经过在哈尔滨工程大学电磁场与微波专业的教学与实践,得到了学生广泛好评,取得了良好的教学效果。

参加该书翻译的有哈尔滨工程大学余文华、李文兴、张朝柱、段瀚林、孙照清、李千、卓庆坤、王美玉、张文、刘鑫博、郭晓亮、田春胜、赵月明、刘玉敬、张兰超、孔媛媛、曹小东。该书由国防科技大学刘培国教授校对。

该书适合作为电磁场与微波技术专业高年级本科生和研究生的教材或参考书,也可以作为年轻教师和工程师的参考书。该书为哈尔滨工程大学研究生规划教材。

余文华

于哈尔滨工程大学

致中国读者

This book grew out of a set of notes I prepared for a graduate-level course on the finite-difference time-domain (FDTD) method I taught at Washington State University. The notes were intended to supplement the textbook I used for the course, *Computational Electrodynamics: The Finite-Difference Time-Domain Method* by Taflov and Hagness (3rd ed., Artech House, 2005). The goal of the course was to provide students not only with a thorough understanding of the FDTD method, but also to ensure they were able to implement functional FDTD code and to tackle meaningful, challenging, and interesting problems. (For example, one of the homework assignments in the class was to duplicate the patch antenna modeling in David Sheen's groundbreaking 1991 MIT Ph.D. dissertation.) Each time I taught the course I added to the notes to the point that students believed the assigned textbook should be considered optional, i.e., the material I provided stood on its own as a complete text in and of itself. I have shared the English version of the course material widely and have been flattered by the response. From the feedback I have received, the material has served individuals interested in learning the FDTD method and writing functional code extremely well.

Although this material was used in a graduate-level course, undergraduate students often took the course. They only needed to be sufficiently motivated, to have had an introductory course in electromagnetics and, hopefully, to have had some programming experience. I truly feel that by understanding the FDTD method one can better understand Maxwell's equations and electromagnetics in the continuous world. Thus, I can easily envision this material serving as the textbook for an upper-division undergraduate course.

The FDTD method is computationally expensive. Many interesting simulations can be quickly and easily realized in languages such as MATLAB or Java, but to get the most from the technique it is necessary to code in a computationally efficient language such as FORTRAN or C. In this book, C is the language of choice. Although C may be considered arcane or archaic by some, as used here it is quite accessible—readers need not have much, or perhaps even any, familiarity with C. The FDTD method, at its core, is an extremely simple technique and if you are able to implement a FOR-loop, you have almost all the coding skills you need to do meaningful simulations (a slight exaggeration, perhaps, but as you will see, only a slight one).

Finally, the Chinese edition of this material would not exist without the work of Prof. Wenhua Yu and his students. Thus, I would like to express my deep appreciation to Prof. Yu for reaching out to me and to all those who made the translation of this material possible.

John Brand Schneider
Washington State University

目 录

第 1 章	数字伪像	1
1.1	引言	1
1.2	有限精确度	1
1.3	符号处理	5
第 2 章	电磁学的简要回顾	6
2.1	引言	6
2.2	库仑定律和电场强度	6
2.3	电位移矢量	7
2.4	静电场	9
2.5	梯度、散度和旋度	10
2.6	拉普拉斯运算	12
2.7	高斯定理和斯托克斯定理	15
2.8	电场边界条件	15
2.9	电导率和理想电导体	16
2.10	磁场	16
2.11	磁场边界条件	18
2.12	静态场简述	18
2.13	时域场	18
2.14	时变场总结	19
2.15	无源区的波动方程	20
2.16	一维波动方程的解决方案	20
第 3 章	一维时域有限差分 (FDTD) 法简介	22
3.1	引言	22
3.2	Yee 算法	23
3.3	一维递推方程	23
3.4	一维 FDTD 仿真的计算机实现	27
3.5	基本仿真	29
3.6	一维 PMC 边界条件	31
3.7	场的记录	32
3.8	附加源	35
3.9	边界条件	38
3.10	总场/散射场边界条件	39
3.11	非均匀媒质	44
3.12	色散媒质	49

第 4 章 FDTD 程序优化	56
4.1 引言	56
4.2 数组和动态内存分配	56
4.3 宏	57
4.4 结构体	60
4.5 改进一	65
4.6 模块化设计和函数初始化	69
4.7 改进二	73
4.8 编译模块化代码	78
4.9 改进三	79
第 5 章 推广 FDTD 仿真至所有频率	88
5.1 引言	88
5.2 源	88
5.2.1 高斯脉冲	88
5.2.2 谐波源	88
5.2.3 Ricker 小波	90
5.3 映射频率至离散傅里叶变换	91
5.4 运行离散傅里叶变换 (DFT)	93
5.5 实信号与 DFT 信号	95
5.6 两个时域采样点的振幅和相位	98
5.7 电导率	100
5.8 实例: 获取平面交界面的透射系数	104
5.8.1 穿过平面交界面的透射 (连续空间)	104
5.8.2 使用 FDTD 测量透射系数	106
第 6 章 基于微分方程的吸收边界条件	113
6.1 引言	113
6.2 传播方程	113
6.3 截断网格	114
6.4 一阶吸收边界条件的实现	116
6.5 用算子符号表示吸收边界条件	120
6.6 二阶吸收边界条件	121
6.7 二阶吸收边界条件的实现	123
第 7 章 色散、阻抗、反射和透射	126
7.1 引言	126
7.2 连续空间中的色散	126
7.3 FDTD 方法的谐波表示	127
7.4 FDTD 的网格色散	129

7.5	数值阻抗	132
7.6	FDTD 解析反射系数和透射系数	133
7.7	理想电导体的反射	136
7.8	交界面与电场节点重合	138
第 8 章	二维 FDTD 仿真	142
8.1	引言	142
8.2	多维数组	142
8.3	二维: TM^z 极化	145
8.4	TM^z 实例	148
8.5	TM^z 极化波的 TFSF 边界	160
8.6	TM^z TFSF 边界举例	163
8.7	TE^z 极化	173
8.8	TM^z 和 TM^z 仿真中的 PEC	177
8.9	TE^z 举例	178
第 9 章	三维 FDTD	191
9.1	引言	191
9.2	C 语言中的三维数组	191
9.3	支配方程和三维网格	193
9.4	三维程序实例	199
9.5	TFSF 边界	213
9.6	TFSF 示范	218
9.7	非等空间步长	225
第 10 章	色散材料	229
10.1	引言	229
10.2	本构关系和色散媒质	229
10.2.1	Drude 材料	231
10.2.2	Lorentz 材料	232
10.2.3	Debye 材料	232
10.3	Debye 材料运用辅助差分方程方法 (ADE)	233
10.4	Drude 材料运用辅助差分方程方法 (ADE)	235
10.5	磁色散材料	237
10.6	分段线性递归卷积	239
10.7	Debye 材料的分段线性递归卷积方法	242
第 11 章	完全匹配层 (PML)	245
11.1	引言	245
11.2	有耗层, 一维	245

11.3	有耗层, 二维	247
11.4	分裂场 PML	249
11.5	非分裂场 PML	252
11.6	非分裂场 PML 的 FDTD 实现	255
第 12 章	声学 FDTD 仿真	259
12.1	引言	259
12.2	FDTD 支配方程	260
12.3	二维 FDTD 实现	263
第 13 章	并行处理	265
13.1	线程	265
13.2	线程举例	266
13.3	消息传递接口	272
13.4	Open MPI 基础知识	273
13.5	等级和大小	275
13.6	进程间的通信	276
第 14 章	近远场变换	281
14.1	引言	281
14.2	等效原理	281
14.3	矢量位	282
14.4	远区的电场	288
14.5	复化辛普森积分	291
14.6	电磁场的排列: 几何平均法	291
14.7	几何平均法实现近远场变换	294
14.7.1	双缝辐射	294
14.7.2	圆柱体散射	296
14.7.3	球体前向强散射	298
附录 A	四阶中心差分的构造	302
附录 B	产生瀑布图和动画	303
附录 C	二维数据的渲染及动画	306
附录 D	符号	308
附录 E	PostScript 简介	309
E.1	引言	309
E.2	PostScript 文件	309
E.3	PostScript 基本命令	310
索引		319

第1章 数字伪像

1.1 引言

事实上, 解决几乎所有电磁学问题都需要一台计算机。一个理想问题即使有准确的解析解或闭式解, 对于给定的一组参数, 我们也必须使用计算机将解析解转变为数值解。由于数据在计算机内存储方式的固有限制, 计算结果总是不可避免地具有一定的误差, 虽然这些误差通常都很小, 但是我们应该知道这一现象。首先, 我们将讨论一些关于有限精度产生的后果。

稍后, 我们将讨论基于时域有限差分 (FDTD) 方法的电磁问题数值解。FDTD 方法采用的近似技术使其得到的解也是近似的, 也就是说这个方法本质上是近似方法, 即使我们使用计算机可以获得无限精度的数值解, 用 FDTD 方法得到的结果仍是一个近似解。在后面的章节中将讨论在 FDTD 方法中的近似问题。

使用数值方法时需要注意一点, 如果数值方法在求解过程中不发生颠覆性错误, 计算机总是会给出一个结果。我们可能经常发现运行程序很简单, 调试程序的过程却非常复杂。当程序能够运行时, 通常情况下假设所有的错误都已被纠正了, 可惜的是, 事情常常并非如此。让程序运行是一回事, 得到正确的结果是另一回事, 事实上, 得到精确的结果又是另外一回事——解可能只对给定的情况是正确的, 但是求解过程可能得不到足够精确的结果。因此, 最好是采用更多的方式检验实现过程和结果。例如, 可以通过一种分辨率得到一个解, 使用更高的分辨率又可能得到另一个不同的解, 如果这两个解不是很接近, 至少解还没有收敛到“正确的”解, 就必须使用更高的分辨率, 或许在解的过程中有系统错误存在。基本事实是: 计算机给出一个结果, 但并不意味着这个结果一定正确。

1.2 有限精确度

众所周知, 3 个 $\frac{1}{3}$ 之和等于 1, 即 $1/3 + 1/3 + 1/3 = 1$ 。但是, 在计算机上做这样的运算正确吗? 考虑程序 1.1 所示的 C 程序。

在这个程序中单精度变量 a 设置为 $\frac{1}{3}$ 。在第 9 行, 3 个 a 之和与实数 1.0 进行比较, 如果它们相等, 程序输出 “Equal”, 否则输出 “Not equal”。这个程序的输出是 “Not equal”, 因此, 对于计算机 (至少可以运行一种常用的求解电磁学问题的语言) 来说, $1/3 + 1/3 + 1/3$ 不等于 1。值得注意的是如果把第 9 行写成 $a = 1/3$, 因为使用整数运算法则求解除法的值, a 将会被置零。当用 $a = 1.0/3.0$ 表示时, 计算机就会使用浮点型法则运算。

程序 1.1 oneThird.c: 测试 $1/3 + 1/3 + 1/3$ 是否等于 1。

```

1  /* 1.0/3.0 + 1.0/3.0 + 1.0/3.0 == 1.0? */
2  #include <stdio.h>
3
4  int main() {
5      float a;
6
7      a = 1.0 / 3.0;
8
9      if (a + a + a == 1.0)
10         printf("Equal.\n");
11     else
12         printf("Not equal.\n");
13
14     return 0;
15 }

```

在 C 语言或 FORTRAN 语言中浮点型数据类型只能存储一个数的有限位。在大部分的机器中,单精度数用 4 个字节(32 个二进制位或比特)存储,双精度数用 8 个字节(64 位)存储。再次考虑 3 个 $\frac{1}{3}$ 求和问题,举一个极端的例子,假设计算机只能存储两位小数, $\frac{1}{3}$ 约等于 0.33。把这个数求和 3 次得到

$$0.33 + 0.33 + 0.33 = 0.99$$

很明显结果不等于 1。如果存储数的位数更多,结果将更接近于 1,但永远不等于 1。十进制和二进制浮点型所表示的 $\frac{1}{3}$ 都有无限的位数。因此,当我们试图把 $\frac{1}{3}$ 存储到计算机时数字会被截断,所以计算机只存储 $\frac{1}{3}$ 的近似值。因此对被截断的 $\frac{1}{3}$ 求和 3 次的结果不等于 1。

如果我们对 $\frac{1}{10}$ 求和 10 次,结果是多少呢?因为 $1/10$ 等于 0.1,似乎这个数可以用有限数量的位数来存储。计算机认为下面的等式成立吗?

$$\frac{1}{10} + \frac{1}{10} + \frac{1}{10} + \frac{1}{10} + \frac{1}{10} + \frac{1}{10} + \frac{1}{10} + \frac{1}{10} + \frac{1}{10} + \frac{1}{10} = 1?$$

可以用同样的方法在程序 1.1 中进行验证。奇怪的是,结果仍然不等于 1! 虽然用十进制表示 $\frac{1}{10}$ 时具有有限的位数,但是当用二进制表示时就有无限的位数。在浮点型十进制数中每一位都表示 10 的一个特定幂数。令每一个空格表示一位,一个数可以用下面的方式理解:

$$\dots \quad \underline{\hspace{1cm}} \quad \underline{\hspace{1cm}} \quad \underline{\hspace{1cm}} \quad \underline{\hspace{1cm}} \quad \cdot \quad \underline{\hspace{1cm}} \quad \underline{\hspace{1cm}} \quad \underline{\hspace{1cm}} \quad \underline{\hspace{1cm}} \quad \dots$$

$$10^3 \quad 10^2 \quad 10^1 \quad 10^0 \quad 10^{-1} \quad 10^{-2} \quad 10^{-3} \quad 10^{-4}$$

每一位可以告诉我们它对应 10 的幂数是多少,小数点是负指数和非负指数的分界线。与十进制相似,二进制中的一位表示 2 的幂数:

$$\dots \quad \overline{\quad} \quad \overline{\quad} \quad \overline{\quad} \quad \overline{\quad} \quad \cdot \quad \overline{\quad} \quad \overline{\quad} \quad \overline{\quad} \quad \overline{\quad} \quad \dots$$

$$2^3 \quad 2^2 \quad 2^1 \quad 2^0 \quad 2^{-1} \quad 2^{-2} \quad 2^{-3} \quad 2^{-4}$$

十进制数 0.1_{10} 就是 1×10^{-1} 。为了利用二进制数获得相同的值, 必须取 $2^{-4} + 2^{-5} + 2^{-8} + 2^{-9} + \dots$, 即无限位数的二进制数。另一种表示方法是

$$0.1_{10} = 0.0001100110011001100110011\dots_2$$

如前所述, 当这个数存储在计算机中时, 数字被截断, 所存储的值不再正好等于 $\frac{1}{10}$, 对这个值 10 次求和就不等于 1 (尽管差别非常小)。如何将浮点型值存储在计算机中不是我们关心的主要问题, 然而, 了解比特如何分配是有益的。数字以指数形式存储, 比特的标准分配是:

	总比特数	符号	尾数	指数
单精度	32	1	23	8
双精度	64	1	52	11

本质上, 指数给出一个数的量级, 尾数给出一个数的位数——尾数决定精度。尾数的位数越多, 表示的数越精确。双精度的位数是单精度的两倍, 它的尾数有 52 位而单精度只有 23 位, 因此, 双精度数的精度是单精度的两倍多。尾数为 23 位的二进制数比 7 位十进制数略小, 这是因为 2^{23} 是 8,388,608, 因此 23 位二进制数可以表示 0 和 8,388,607 之间的数。换句话说, 尾数为 52 位的二进制数所表示的值对应于十进制数 15 和 16 位之间 ($2^{52} = 4,503,599,627,370,496$)。对于指数, 一个双精度数的指数位数比单精度数多 3 位。看起来似乎双精度的指数变化不大, 因为它的位数不是单精度的指数位数的两倍。然而, 指数表示的是一个数的大小, 每多一比特从本质上就表示数值加倍。如果指数的位数为 9, 它可以表示两倍于单精度数的数。一个双精度数所拥有的 3 位额外位数表示的指数比单精度数大 8 倍, 即可以转变为比单精度数大 (或小) 256 倍的数。考虑下面的方程:

$$a + b = a$$

从数学的角度分析, 我们知道当 b 等于 0 时这个方程才满足。然而, 当使用计算机时这个方程可以成立, 也就是说, b 对 a 没有任何贡献, 即使 b 为非零值。

当一些数相加或相减时, 它们的尾数就会改变直到它们的指数相等。此时, 尾数可以直接相加或相减。然而, 如果指数间的差别大于尾数的长度, 当加上或从较大的数中减去较小的数时不会有任何影响。程序 1.2 中的代码说明了这一现象。

这里 a 的初始值为 1.0, b 设定为 0.5。变量 c 为 a 和 b 之和。While 循环开始于第 5 行, 只要 c 不等于 a , 将一直循环下去。在循环体中, b 被 2 除, c 再次等于 a 加 b 。如果计算机有无限的精度, 这将会是一个无限循环。 b 的值将趋向于无穷小, 但是永远不会等于 0, 因此 a 加 b 永远不可能等于 a 。然而, 循环会终止, 在第 10 行 `printf()` 打印输出为

```
1 5.96046e-08 1
```

这说明当 b 的值为 5.96046×10^{-8} 时, a 和 c 是一致的。注意到 b 的这个值对应于 1×2^{-24} , 当 b 等于这个值时, a 和 b 指数的差别将大于 23 (a 是 1×2^0)。

程序 1.2 测试一个非零数 b 是否满足方程 $a + b = a$ 。

```
1  float a = 1.0, b = 0.5, c;  
2  
3  c = a + b;  
4  
5  while(c != a) {  
6      b = b / 2.0;  
7      c = a + b;  
8  }  
9  
10 printf("%12g %12g %12g\n", a, b, c);
```

再举一个例子来说明有限精度不以显著的方式使计算结果变差。假设变量 a 等于 2，对 a 取平方根，再对其值进行平方运算，得到的结果应该接近于 2（理想情况下是 2，但是由于 $\sqrt{2}$ 有无限个位数，精度上将会有一些丢失）。然而，如果对数字 2 取 23 次平方根后再对结果进行 23 次平方会发生什么呢？我们希望得到的结果接近于 2，但事实并非如此。程序 1.3 所示的程序将帮助我们验证这种情况。

程序 1.3 rootTest.c: 平方根测试。

```
1  /* 平方根测试 */  
2  #include <math.h> // 为了sqrt()函数  
3  #include <stdio.h>  
4  
5  #define COUNT 23  
6  
7  int main() {  
8      float a = 2.0;  
9      int i;  
10  
11     for (i = 0; i < COUNT; i++)  
12         a = sqrt(a); // a 的平方根  
13  
14     for (i = 0; i < COUNT; i++)  
15         a = a * a; // a 的平方  
16  
17     printf("%12g\n", a);  
18  
19     return 0;  
20 }
```

程序的输出是 1，即结果是 $a=1.0$ 。每进行一次平方根，得到的值越来越接近 1。最后，由于截断误差，计算机认为值是 1。这时，即使对它进行再多的平方也不会改变其大小。

1.3 符号处理

使用常用的数值分析语言（如 C、C++、FORTRAN 或者 MATLAB）时，截断误差是不可避免的。圆周长与半径的比值为 $\pi = 3.141592\dots$ ，这是一个无限循环的无理数，这个数不可能在计算机内存储 π 的准确数值，相反地，我们必须存储一个具有有限位的近似数。然而，有些软件（例如 Mathematica）允许对数学符号运算。在 Mathematica 软件中，你可以将 π 写成 `Pi`，Mathematica 软件知道这个数学符号代表什么。例如，`10000000001*Pi` 的余弦值等于 `-1`。相似地，也可以写成 `Sqrt[2]`，Mathematica 软件知道它的平方恒等于 2。只可惜这种符号运算在计算资源方面是非常昂贵的，许多电磁学的前沿问题包含成千上万甚至上百万的未知量。为了处理这些大量的数据，数据运算在内存和时间上必须尽可能地高效。Mathematica 软件虽然是非常好的工具，但它并不是解决大型数值问题的有效工具。

在 MATLAB 中我们可以把 `pi` 看作 π 的缩写。然而，这里所表示的 π 和 Mathematica 软件中用到的并不一样。在 MATLAB 中，`pi` 在本质上只是数值表示——它只是比写成完整的 `pi` 方便。在 C 语言中，将会提供包含 `math.h` 的头文件，可以用 `M_PI` 作为 π 的缩写。在 `math.h` 中可以看到下面的语句：

```
# define M_PI          3.14159265358979323846  /* pi */
```

这恰好和 MATLAB 中的一样，MATLAB 只知道 `pi` 的数值，而且这只是正确值的一个截断值。因此，将会用 `10000000001*pi` 的余弦值等于 `-0.9999999999954` 来代替准确值 `-1`（当然在这种情况下误差是微不足道的）。

第2章 电磁学的简要回顾

2.1 引言

我们将在后面的章节中详述与时域有限差分(FDTD)方法相关的特殊方程组。本章回顾一下以前在电磁学课程中所学过方程组的物理意义。

从某种程度上说,有几个简单规律可作为所有电磁学问题的前提。电磁学是基于下面事实(有人可能不同意):

- (1) 一个电荷作用力到另外一个电荷上。
- (2) 一个运动的电荷作用力到其他运动电荷上。
- (3) 所有物质都是由带电粒子组成的。

当然,把这些规律转换为相应的数学表达式并不是很简单的。然而,我们不应忽略的事实是,数学正试图描述概念相当简单的原理。

2.2 库仑定律和电场强度

库仑(Coulomb)研究了带电粒子间所受的电场力。如图2.1所示,两个分离点电荷所带的电荷分别为 Q_1 和 Q_2 ,点电荷 Q_2 受到点电荷 Q_1 的作用力并且力的方向沿着两点间的直线。

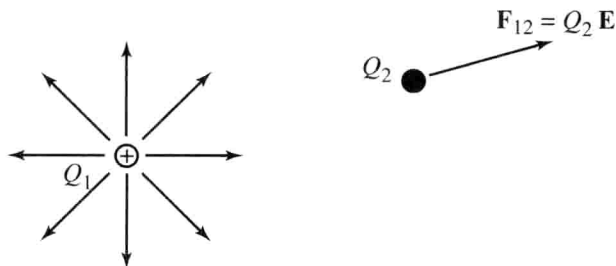


图 2.1 电荷之间的电场力

电荷 Q_2 受到电荷 Q_1 的力沿两电荷间的直线方向,力的方向由两电荷的正负决定。电场指向径向远离正电荷的方向,如用指向远离 Q_1 的直线表示(这里假设 Q_1 是正电荷)

电场力与电荷量成正比,与电荷之间距离的平方成反比。电荷 Q_2 受到电荷 Q_1 的作用力可以表示为

$$\mathbf{F}_{12} = \hat{\mathbf{a}}_{12} \frac{1}{4\pi\epsilon_0} \frac{Q_1 Q_2}{R_{12}^2} \quad (2.1)^*$$

* 本书的符号用法保持英文原书的风格。

式中 $\hat{\mathbf{a}}_{12}$ 为从电荷 Q_1 指向电荷 Q_2 的单位矢量, R_{12} 是电荷间的距离, $1/4\pi\epsilon_0$ 是常数。 ϵ_0 是真空的介电常数, 大小约为 8.854×10^{-12} F/m。电荷的单位是库仑 (C), 电荷有正负之分。当两电荷电性相同时, 两电荷间相互排斥: $\mathbf{F}_{12}$ 的方向与 $\hat{\mathbf{a}}_{12}$ 的方向相同。当两电荷电性相反时, $\mathbf{F}_{12}$ 的方向与 $\hat{\mathbf{a}}_{12}$ 的方向相反。

方程 (2.1) 有一个不足, 因为它意味着力作用于一段距离。这个方程似乎告诉我们相互作用力 $\mathbf{F}_{12}$ 的建立是即时的, 从这个方程可以假定电荷之间距离 R_{12} 的改变将导致作用力 $\mathbf{F}_{12}$ 的瞬间改变, 但事实并非如此。一个电荷位置的变化传达到另一个位置的电荷需要有限的时间间隔 (类似地, 电荷量的变化传达到另一个电荷也要花有限的时间)。为了克服这个不足, 利用场的概念解释是很方便的。我们不说 Q_1 直接作用于 Q_2 上, 而是理解为 Q_1 产生了一个场, 这个场产生了一个作用于 Q_2 的力。 Q_1 产生的场不依赖于 Q_2 ——无论 Q_2 是否存在并感受到它, Q_1 的场都一直存在。

在静态情况下, 场的方法并不比直接利用库仑定律更有优势。这是因为对于静电荷, 库仑定律是正确的。为了能够区别, 场是时变的。不过, 与时变情况下一致, 场同样适用于静态情况。点电荷 Q_1 产生的电场强度为

$$\mathbf{E}_1 = \hat{\mathbf{a}}_r \frac{Q_1}{4\pi\epsilon_0 r^2} \quad (2.2)$$

其中 $\hat{\mathbf{a}}_r$ 为径向远离点电荷的单位向量, r 为空间中任意一点到点电荷的距离。电场强度的单位为伏特每米 (V/m)。

为了得到作用于 Q_2 上的力, 仅需要将其电荷乘以电场强度: $\mathbf{F}_{12} = Q_2 \mathbf{E}_1$ 。一般地, 任意电荷所受的力为该电荷与该点电场强度的乘积, 也就是 $\mathbf{F} = Q\mathbf{E}$ 。

2.3 电位移矢量

所有物质都是由带电粒子组成的, 物质可能不显示电性是因为它带有等量的正负电荷。然而, 在包括电场影响在内的多种情况下, 物质内部的正负电荷可能发生轻微偏移, 由此造成的电荷分离将对整个电场产生影响。正因为如此, 引入电位移矢量 $\mathbf{D}$ 是非常方便的, 电位移矢量 $\mathbf{D}$ 的单位是库仑每平方米 (C/m²), 本质上 $\mathbf{D}$ 忽略了约束在物质内部电荷局部影响。在自由空间中, 电场强度和电位移矢量的关系是

$$\mathbf{D} = \epsilon_0 \mathbf{E} \quad (2.3)$$

高斯定律表明 $\mathbf{D}$ 在闭合曲面的积分等于该闭合曲面包围的自由电荷量, 即

$$\oint_S \mathbf{D} \cdot d\mathbf{s} = Q_{\text{enc}} \quad (2.4)$$

其中 S 是闭合曲面, $d\mathbf{s}$ 是一个面元增量且它的法线垂直向外, Q_{enc} 是闭合曲面包围电荷的电荷量。例如, 考虑方程 (2.2) 给定的电场强度, 假定 S 是一个球面且电荷在其中心, 在方程 (2.4) 中进行简单积分:

$$\oint_S \mathbf{D} \cdot d\mathbf{s} = \int_{\theta=0}^{\pi} \int_{\phi=0}^{2\pi} \epsilon_0 \frac{Q_1}{4\pi\epsilon_0 r^2} \hat{\mathbf{a}}_r \cdot \hat{\mathbf{a}}_r r^2 \sin\theta d\phi d\theta = Q_1 \quad (2.5)$$

实际上其结果并不依赖于所选择的积分曲面（假设它包围所有的电荷），但是对于球面来说，积分运算特别简单。

我们希望方程 (2.4) 中的积分结果和在自由空间中一样总是等于包围的电荷量。然而，当物体存在时，事情就变得比较复杂。如图 2.2 所示，考虑具有等量且均匀电荷分布的两个大平行板，但电荷的符号相反。

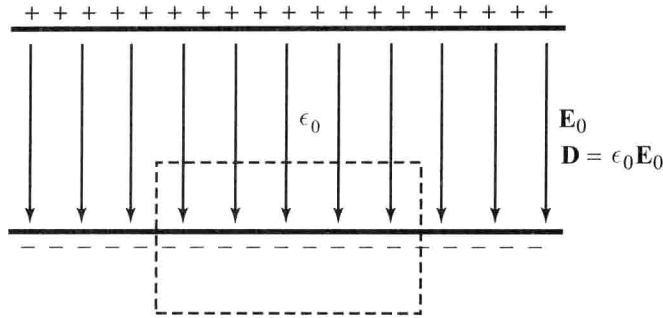


图 2.2 自由空间中的带电平行板
虚线表示积分表面 S

虚线表示一个距离平板边缘足够远的积分表面 S ，使得电场在表面 S 上均匀分布。设电场强度恒等于 E_0 ，平行板外的电场强度为 0，平行板内的电场强度和 S 的侧边相切。因此对积分有贡献的只是 S 的顶部。积分的结果 $\oint_S \epsilon_0 \mathbf{E} \cdot d\mathbf{s}$ 是由表面 S 包围的负电荷（也就是说，负电荷位于表面 S 内部平行板的底部）。

现在考虑同样的平行板，带有同样的电荷，但是在平行板内充满介质。假设这个介质是“可极化的”，介质内的正负电荷可以略微偏移，但电荷不能完全自由移动——它们是被束缚的电荷。正电荷被上面的板排斥，被下面的板吸引。相反地，负电荷被下面的板排斥，被上面的板吸引，如图 2.3 所示。

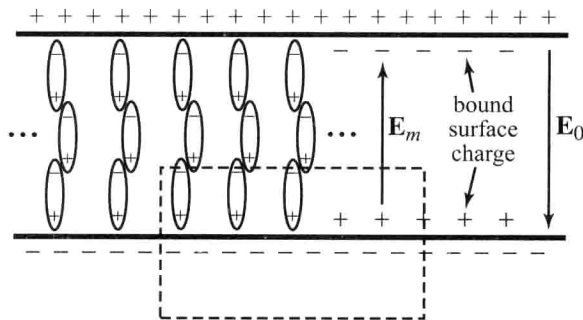


图 2.3 在带电平行板之间有一个可极化介质

细长的椭圆表示分子，它们的定向电荷在上板的下面产生一个负电荷层，在下板的上面产生一个正电荷层。在介质内部，正负束缚电荷相互抵消。在物体的表面必须考虑束缚电荷。因此，在整个图中，分子没有全部画出来。如图 2.3 右边所示，仅仅束缚电荷层被表示出来。平行板上的自由电荷形成的电场强度为 E_0 。束缚电荷形成了和 E_0 反向的电场强度 E_m ，因此使总电场强度变小。虚线仍然表示积分表面 S

尽管介质的存在，平行板上电荷所产生的电场强度仍然是 E_0 ，也就是和图 2.2 中的电场强度一样。然而，在两平行板之间由于极化介质上束缚电荷的偏移存在另外一个电场。