

 MATLAB®&Simulink®
开发实例系列丛书

在线交流，有问有答

(详见封底勒口)



书中所有程序的源代码
可通过扫描此二维码
免费下载。

MATLAB向量化编程 基础精讲

马良 祁彬彬 编著



北京航空航天大学出版社
BEIHANG UNIVERSITY PRESS

MATLAB®
examples

MATLAB®&Simulink® 开发实例系列丛书

MATLAB 向量化编程基础精讲

马 良 祁彬彬 编著



下载程序请用
QQ 浏览器扫码

北京航空航天大学出版社

内 容 简 介

本书使用 MATLAB 最新版本 2016a, 拣选 Mathworks 官方群组 Cody 中一些有趣的代码问题, 分 6 章讲解这些优秀示例代码中使用数组、字符串操作、正则表达式以及匿名函数等方面的 MATLAB 编程技巧, 并对其中较为典型和精彩的用法做扼要点评, 对一些复杂思路或代码的细节和步骤, 还逐一展开了延伸分析, 使学习 MATLAB 编程的用户, 能迅速体会 MATLAB 矢量化编程语言的基本特色。

本书适合所有 MATLAB 编程爱好者和使用 MATLAB 的不同专业大学生阅读, 还可供研究生、科研工作人员及高校教师参考。

图书在版编目 (CIP) 数据

MATLAB 向量化编程基础精讲 / 马良, 祁彬彬编著

北京: 北京航空航天大学出版社, 2016.8

ISBN 978-7-5124-2209-4

I. ①M… II. ①马… ②祁… III. ①Matlab 软件—程序设计 IV. ①TP317

中国版本图书馆 CIP 数据核字 (2016) 第 186538 号

版权所有, 侵权必究。



MATLAB 向量化编程基础精讲

马 良 祁彬彬 编著

责任编辑 王 实

*

北京航空航天大学出版社出版发行

北京市海淀区学院路 37 号 (邮编 100191) <http://www.buaapress.com.cn>

发行部电话: (010)82317024 传真: (010)82328026

读者信箱: goodtextbook@126.com 邮购电话: (010)82316936

北京兴华昌盛印刷有限公司印装 各地书店经销

*

开本: 787 × 1092 1/16 印张: 27.75 字数: 710 千字

2017 年 3 月第 1 版 2017 年 3 月第 1 次印刷 印数: 4 000 册

ISBN 978-7-5124-2209-4 定价: 58.00 元

若本书有倒页、脱页、缺页等印装质量问题, 请与本社发行部联系调换。联系电话: (010)82317024

前 言

学习 MATLAB，从来不是“学习 MATLAB”这么简单。

从一开始，对它的学习就和所学专业领域的相关理论同步，在学习阶段对它们的理解又交错生长、相互促进。毫无疑问，专业问题的研究处于核心主体地位，它高于对一个具体工具软件的钻研，但我们往往需要让公式、语言描述等，能以 MATLAB 作为媒介，解释、模拟、甚至预测事物运转的规律和真相。但这对于多数未必见长于编程的工程师，或者非计算机专业的高校大学生，具有一定的挑战性。

所以这时，学习方法就显得更加重要，人常说“工欲善其事，必先利其器”，可遇到的麻烦却往往是“器利，工未取之以确法，致事不善”。层出不穷、匪夷所思的代码问题，往往是学习 MATLAB 伊始，没养成良好的编程习惯、没按正确方法发挥 MATLAB 特点所致。“良好习惯”或“正确方法”，并不仅仅是“每行代码都加注释”、“写一行隔个空行”等，当然，良好的编程习惯对代码后期维护调试大有好处，但这不是本书重点探讨的问题。我们要说的是：深入了解乃至掌控 MATLAB 函数，达到有效、简捷地用代码解决问题之目标。要达到这样的程度，恐怕要从调用方式到搭配组合再到执行效率，完整透彻理解 MATLAB 一些常用函数命令后，才能做到。很多人以为不难，认为看看命令帮助，学几个常见调用格式，写出程序，没有红色出错警示，就算大功告成了。

真是这样吗？

举例而言：其实相当一部分用过 MATLAB 软件，哪怕使用多年的用户，对 MATLAB 的常用命令也都未必谈得上熟悉。不信？不妨试试下面这个对带有“非数”的数列求和的问题：

源代码 1: 带有非数时的求和

```
1 >> a=[1:5,NaN,7]
2 a =
3     1     2     3     4     5 NaN     7
4 >> sum(a)
5 ans =
6     NaN
```

源代码 1 说明，当元素序列中存在特殊元素“NaN”时，原有的代数运算规则将发生变化，比如： $\text{NaN}+1=\text{NaN}$ ， $\text{NaN}+\text{inf}=\text{NaN}$ (NaN 的详细介绍见 1.11.2 小节)。但在实际运算中这往往没有意义，我们可能更多需要的是统计除“NaN”之外的其他元素之和。

很多人想到循环遍历判断每个元素是否为“NaN”：

源代码 2: 除“NaN”以外元素的求和——方法 1

```
1 for i=1:length(a)
2     if isnan(a(i))
3         a(i)=0;
4     end
```

```
5 end
6 Result=sum(a)
```

源代码 2 用循环遍历序列 a 的每个元素，通过命令 `isnan` 判断每个元素是否为“NaN”，如果是用 0 替换，最后求和。

对于没怎么接触过 MATLAB 的读者而言，源代码 2 貌似不错：一个程序用到循环、判断两种流程，甚至还有 `isnan` 这样“高端大气上档次”的逻辑命令。但更加了解 MATLAB 向量化操作的用户都知道，函数 `isnan` 支持向量化逻辑操作，循环、判断流程可以全部去掉。

源代码 3: 除“NaN”以外元素的求和——方法 2

```
1 a=[1:5,NaN,7];
2 Result=sum(a(~isnan(a)))
```

当然，在已知数组 a 确定为正的情况下，`isnan` 可用大于零的逻辑判断： $a(a >= 0)$ 代替，这是针对具体问题的特殊构造。

到此，即使具有一定 MATLAB 使用经验的读者，可能都会认为已经简无可简了，但重读求和命令 `sum` 后，你会发现 MATLAB 给这个使用频率最高的函数，悄然加上了后置辨识参数“`nanflag`”，专门用于判定数组或者矩阵求和过程是否应当略过“非数”。它有两个选项：“`{'includenan'} | 'omitnan'`”，花括号内的是默认值，这也是为什么直接对数组 a 求和而得到的结果却是“NaN”的原因，所以用 `sum` 求和时，把“`nanflag`”后置识别参数换为第二项，也就是“`'omitnan'`”，可直接得解。

源代码 4: 除“NaN”以外元素的求和——方法 3

```
1 a=[1:5,NaN,7];
2 Result=sum(a,'omitnan')
```

是不是更简单了呢？我们可以举一反三，不仅求和函数，在 `max`、`min`、`mean`、`std`、`cov` 等不少经常使用的命令中也有类似的“非数”辨识参数选项，有兴趣的话可以在帮助中搜索“`nanflag`”查看更详细的内容。

仍以 `sum` 命令为例，有点基础的读者都知道 MATLAB 中的运算是以列为第一方向的，所以 `sum` 对于矩阵是按列求和的，如果要求按行求和，很多人会习惯性地先转置再求和：

源代码 5: 矩阵按行求和——方法 1

```
1 >> a=randi(10,4)
2 a =
3     9     7    10    10
4    10     1    10     5
5     2     3     2     9
6    10     6    10     2
7 >> sum(a')
8 ans =
9    36    26    16    28
```

但 `sum` 函数中有一个维度指定的后缀参数“`dim`”，就省去了从外部转置的步骤：

源代码 6: 矩阵按行求和——方法 2

```

1 >> sum(a,2)
2 ans =
3     36
4     26
5     16
6     28

```

源代码 6 中通过第 2 个参数指定了求和方向为第 2 维度，即列方向。

一些读者觉得两种方法其实一样，第 2 种方法无非在内部做转置，与单独在外部做转置的方法“殊途同归”。这里要指出的是，两种方法原理上有很大区别：一方面，强调尽可能多运用相对高效的内置函数，能在内部解决的问题尽量不放在函数外部；另一方面，也是更重要的，当矩阵维度进一步扩展时，前一种方法自动失效，比如对三维矩阵 ($m \times n \times l$)，如需按第 3 维度 l 求和，则可深入到元素做遍历循环：

源代码 7: 三维矩阵按“页”求和——方法 1

```

1 a=randi(10,4,4,2);
2 for i=1:size(a,1)
3     for j=1:size(a,2)
4         Result(i,j)=a(i,j,1)+a(i,j,2);
5     end
6 end
7 Result

```

如果知道高低维索引转换命令 `ind2sub` 的用法，则二重循环降至一重也未尝不可：

源代码 8: 三维矩阵按“页”求和——方法 2

```

1 a=randi(10,4,4,2);
2 for i=1:numel(a(:,:,1))
3     [I,J]=ind2sub(size(a(:,:,1)),i);
4     Result(I,J)=a(I,J,1)+a(I,J,2);
5 end
6 Result

```

不过在循环机制下，还是按页整体求和相对直观和高效，毕竟 MATLAB 支持同维矩阵元素的对位相加：

源代码 9: 三维矩阵按“页”求和——方法 3

```

1 a = randi(10,4,4,2);
2 Result = a(:,:,1);
3 for i = 2 : size(a,3)
4     Result = Result + a(:,:,i);
5 end
6 Result

```

若对多维矩阵操作命令有一定基础，则把数据按问题要求变维再求和也能达到要求：

源代码 10: 三维矩阵按“页”求和——方法 4

```

1 squeeze(sum(permute(a,[3,2,1])))'

```

在源代码 10 中, 按照 `sum` 的求和顺序, 先用 `permute` 重排多维数组求和, 再用 `squeeze` 压缩多维矩阵还原为结果。

上述对多维矩阵在高维度上的求和, 明显感到循环遍历元素、变维等办法都很繁琐, 其实只要更改 `sum` 默认维度参数 “`dim`”, 源代码 7~10 遇到的问题就都能避免:

源代码 11: 三维矩阵按 “页” 求和——方法 5

```
1 sum(a,3)
```

如果对 MATLAB 的 `cell` 数据结构理解更多一些, 则会发现一些涉及 `cell` 数据结构的命令也具有数据打乱重组的方式, 求和则可通过 `cellfun` 函数调用求和句柄对归并数据完成操控:

源代码 12: 三维矩阵按 “页” 求和——方法 6

```
1 cellfun(@sum,num2cell(a,3))
```

以上是求和命令 `sum` 的应用示例, 此外, 分析时间序列的工具箱 (Financial Toolbox) 函数 `nansum` 同样可以指定维度, 并自动忽略数据中的 “NaN” 求和, 感兴趣的读者可在 “帮助” 中查看。另外, 如果今后对 MATLAB 函数有了更深入透彻的认识, 涉及数据的重组归并还可参照 `accumarray`、`splitapply` 等函数。

从上述矩阵求和例子能看出: 一方面, 掌握 MATLAB 函数是长期累积的过程, 很多甚至是十分常见的命令, 其调用方法也会随版本更替不断 “进化”, 需要不断学习和体会, 并没有一劳永逸的捷径; 另一方面, 不少省时省力的扩展方法也说明, 钻研内置函数是有潜力可挖的。此外, 也建议读者朋友在条件允许的情况下, 尽量使用新版本, 因为每次新版本对一些命令调用格式的微调, 往往给 MATLAB 编程工作带来意想不到的切实便利。

鉴于此, 我们决定尝试总结一些函数综合运用的心得体会, 帮助大家有针对性地训练在 MATLAB 中操控数组和字符串的技巧, 以具体问题为导向, 尽量贴近实战环境, 把复杂问题的运算过程, 分解成多个简单的 “代码步”, 由浅入深, 逐步解释命令的组合与搭配思路, 使问题化繁为简、读者容易理解, 举一反三, 对 MATLAB 命令在具体环境中的用法有更深一层的体悟。

要写出好的代码, 首先要能欣赏好的代码。本书中所选择的问题, 大多来自 Cody (Mathworks 公司主页上一个用 MATLAB 编程解决小问题的社区群体), 在每个问题后, 我们都给出了多种解决代码, 以及关键窍要处的点评和注解, 读者可以通过这些代码, 洞见函数细微处控制的精妙 “杀招”, 开阔代码编写思路。相信打好这个基础, 将为大家今后使用工具箱命令或自编函数, 以高效简捷地解决专业上的具体问题, 节省大量时间和精力! 我想, 随着代码欣赏力的提高, 佐以适当练习, 慢慢地您也能写出优雅如诗的 MATLAB 程序, 到那时您就会发现写 MATLAB 代码解决问题的过程, 居然充满了令人愉快的成就感!

我想, 这就是我们写书的初衷和最终目的。

沟通和交流也是开阔 MATLAB 代码视界的有效途径, 三人行必有我师, 为与读者朋友们方便地交流和互相学习, 本书在 MATLAB 中文论坛专设了交流版面 (网址: <http://www.ilovematlab.cn/forum-260-1.html>), 如果在阅读本书和运行代码过程中, 您有任何问题, 欢迎

来和我们互动讨论。同时，由于时间仓促，水平有限，书中难免有错误和疏漏，如果您发现有任何问题，请在本书的勘误网址（<http://www.ilovematlab.cn/thread-489591-1-1.html>）提出，我们会尽快改正。

最后，我们感谢在本书内容上和求解代码中贡献智慧的 Cody 社区的兄弟姐妹，这些无名英雄默默的努力，正成为后人在黑暗中摸索 MATLAB 技巧的指路明灯。真诚感谢在探索 MATLAB 技巧的十几年的学习过程中，因网络结识的吴鹏、李国栋、谢中华、刘亚龙、黄源、刘鹏、LY Cao 等朋友，以及一直致力于推广 MATLAB 应用的麦客技术联盟，在本书撰写过程中，得到了你们很多宝贵的建议和意见。感谢北京航空航天大学出版社的编辑一直以来的帮助和鼓励。作者马良感谢母亲柳天毅长期的关心照顾，弟弟马强、好友王华和周兆军等一直以来在精神上的鼓励和支持；作者祁彬彬感谢身后一直默默支持自己的爱人邵冰华。同时，对马文涛、韩风霞、张致旭、窦婷、李伟东、安超、宋曦尧、赵昱杰、张国锋、孔祥松、魏志勋、徐浩鹏、丁洋、刘晨、门特、李曼茹、李森、李平、张超、谷翔、郑瑞峰、江海翔、李凯琪、殷凯、富文莲、褚传乐、孙海龙、吕晓龙、郭智鹏、曹璐、刘凯、支铁城等人在平时工作上的支持，也表示衷心的感谢。

马 良

2016 年 4 月于东北大学

目 录

| | | |
|-------|--|----|
| 第 1 章 | 数组操作初步 | 1 |
| 1.1 | 数组基础训练：算盘里的学问 | 1 |
| 1.1.1 | 逐列循环结合正反向搜索 | 3 |
| 1.1.2 | 利用累积乘积函数 <code>cumprod</code> | 6 |
| 1.1.3 | 构造特殊的乘积因子 | 7 |
| 1.2 | 数组基础训练：非零元素赋值为 1 | 8 |
| 1.2.1 | 循环 + 判断 | 8 |
| 1.2.2 | 利用逻辑判断 + 矢量索引 | 9 |
| 1.2.3 | 利用 <code>abs</code> 和 <code>sign</code> | 9 |
| 1.2.4 | <code>min</code> 函数更改 <code>nanflag</code> 设置参数 | 10 |
| 1.3 | 数组基础训练：将指定元素换成 0 | 11 |
| 1.3.1 | 循环 + 判断 | 12 |
| 1.3.2 | 高低维索引转换后赋值 | 13 |
| 1.3.3 | 利用 <code>bsxfun</code> 单一维扩展构造逻辑判断条件 | 15 |
| 1.3.4 | 利用 <code>sparse</code> 函数对全零稀疏矩阵相关元素赋值 | 16 |
| 1.3.5 | 利用累积方式构造向量的 <code>accumarray</code> 函数 | 17 |
| 1.4 | 数组基础训练：正反对角线互换 | 19 |
| 1.4.1 | 寻找元素行列索引关系循环赋值 | 20 |
| 1.4.2 | 利用低维索引查找正反对角元素关系赋值 | 20 |
| 1.4.3 | 结合逻辑数组或点乘构造对角线元素 | 21 |
| 1.4.4 | 利用逻辑“或”操作 | 26 |
| 1.5 | 数组基础训练：寻找真约数 | 28 |
| 1.5.1 | 函数 <code>factor</code> 和组合命令 <code>nchoosek</code> | 28 |
| 1.5.2 | 最大公约数命令 | 29 |
| 1.5.3 | 含求余函数 <code>mod</code> 和 <code>rem</code> 的逻辑判断 | 30 |
| 1.6 | 数组基础训练：康威的《生命游戏》 | 31 |
| 1.6.1 | 枚 举 | 32 |
| 1.6.2 | 循 环 | 34 |
| 1.6.3 | 叠加与卷积 | 35 |
| 1.7 | 数组基础训练：寻找最大尺码的“空盒子” | 40 |

| | | |
|--------------|---------------------------------------|-----------|
| 1.7.1 | 循 环 | 41 |
| 1.7.2 | 利用 conv2 函数 | 42 |
| 1.8 | 数组基础训练: 寻找对角线上的最多连续质数 | 47 |
| 1.8.1 | 卷积命令 | 48 |
| 1.8.2 | 灵活的 max+ diff+ find 函数组合 | 53 |
| 1.9 | 数组基础训练: 扫雷棋盘模拟 | 59 |
| 1.9.1 | 循环遍历元素 + 判断 | 60 |
| 1.9.2 | 构造三对角矩阵的连乘方案 | 62 |
| 1.9.3 | 利用卷积命令 conv2 | 62 |
| 1.10 | 数组基础训练: 移除向量中的 NaN 及其后两个数字 | 65 |
| 1.10.1 | 循 环 | 66 |
| 1.10.2 | 矢量化索引操作 | 67 |
| 1.11 | 数组基础训练: 把 NaN 用左边相邻数字替代 | 70 |
| 1.11.1 | 循环 + 判断 | 70 |
| 1.11.2 | 利用 cumsum 构造符合要求的索引 | 72 |
| 1.12 | 数组基础训练: 涉及类型转换的数据替代 | 75 |
| 1.12.1 | 利用循环判断 | 76 |
| 1.12.2 | cellfun 赋值符合条件的索引位元素 | 77 |
| 1.12.3 | 利用原逻辑索引在 cell 数组中引用赋值 | 77 |
| 1.12.4 | 统一逻辑索引以多输出方式赋值 | 77 |
| 1.13 | 数组基础训练: 递归中的输入输出变量交互 | 79 |
| 1.14 | 小 结 | 81 |
| 第 2 章 | 字符串操作初步 | 82 |
| 2.1 | 字符串基础训练: 字符取反的七种武器 | 82 |
| 2.1.1 | 利用循环 + 判断的传统方式 | 84 |
| 2.1.2 | 矢量化索引与不同函数组合的替换取反 | 85 |
| 2.1.3 | 函数 sprintf+ 逻辑索引构造 | 85 |
| 2.1.4 | 函数 char+ 逻辑数组 + 四则运算符的多种字符串构造方式 | 87 |
| 2.1.5 | 冒号操作做字符格式归并 + ASCII 码值运算转换 | 88 |
| 2.1.6 | 函数 num2str 及其灵活的设定参数 | 90 |
| 2.1.7 | 构造字符向量以输入做逻辑索引取反 | 91 |
| 2.2 | 字符串基础训练: 星号排布 | 92 |
| 2.2.1 | 循 环 | 93 |
| 2.2.2 | 矢量化构造方式 | 95 |
| 2.3 | 字符串基础训练: “开心”的 2013 | 95 |

| | | |
|--------|--|-----|
| 2.3.1 | 循环 + 利用函数 <code>unique</code> 判断 | 96 |
| 2.3.2 | 循环 + <code>num2str</code> 转化年份为字符串分离数字 | 96 |
| 2.3.3 | <code>num2str</code> 分离数字 + 排序做差 | 97 |
| 2.4 | 字符串基础训练: 寻找“轮转”的子字符串 | 99 |
| 2.4.1 | 几种不同的循环方式 | 100 |
| 2.4.2 | 利用卷积命令 <code>conv2</code> + 测试矩阵 | 105 |
| 2.4.3 | 利用 <code>cellfun</code> + <code>strfind</code> + 测试矩阵 <code>gallery</code> | 105 |
| 2.5 | 字符串基础训练: 猜测密码 | 106 |
| 2.5.1 | 循环 + 判断 | 107 |
| 2.5.2 | 矢量化索引方式 | 108 |
| 2.6 | 字符串基础训练: 用指定数量填充字符 | 108 |
| 2.6.1 | 循环判断及 <code>repmat</code> 扩展序列 | 109 |
| 2.6.2 | 利用索引构造扩展 | 110 |
| 2.6.3 | <code>try</code> 流程省略判断 + 函数 <code>strjoin</code> 拼接向量 | 110 |
| 2.6.4 | 利用 2015a 版本中的新函数 <code>repelem</code> | 112 |
| 2.7 | 字符串基础训练: 带判断条件的字符串替代 | 112 |
| 2.7.1 | 循环 + 判断 | 113 |
| 2.7.2 | 矢量化索引构造 | 114 |
| 2.8 | 字符串基础训练: 抽取指定位数数字组成向量并排序 | 116 |
| 2.8.1 | <code>floor</code> + <code>log10</code> + <code>mod</code> 组合 | 117 |
| 2.8.2 | 转换为字符串提取单字符 | 118 |
| 2.9 | 字符串基础训练: 二进制字符中查找最长的“1”序列 | 122 |
| 2.9.1 | 查找逻辑索引做差 | 123 |
| 2.9.2 | 字符匹配方式处理字符串 | 124 |
| 2.9.3 | 查找字符替换为空格 | 125 |
| 2.10 | 字符串基础训练: 剔除指定数字的序列求和 | 126 |
| 2.10.1 | 利用 <code>log10</code> 或 <code>mod</code> 等函数的数值处理 | 126 |
| 2.10.2 | 利用进制转换函数 <code>dec2base</code> | 128 |
| 2.10.3 | 利用数值转字符函数 <code>num2str</code> 构造逻辑索引 | 129 |
| 2.11 | 字符串基础训练: 元胞数组内字符串的合成 | 129 |
| 2.11.1 | 函数 <code>sprintf</code> | 130 |
| 2.11.2 | 利用向量的列排布变维 | 131 |
| 2.11.3 | 函数 <code>strjoin</code> | 132 |
| 2.12 | 小 结 | 133 |

| | |
|---|-----|
| 第 3 章 数组操作进阶：扩维与构造 | 134 |
| 3.1 关于矩阵维数扩充的预备知识 | 135 |
| 3.1.1 repmat 函数 | 135 |
| 3.1.2 索引构造 | 135 |
| 3.1.3 kron 函数扩维 | 136 |
| 3.1.4 meshgrid 和 ndgrid 函数扩维 | 137 |
| 3.1.5 矩阵外积 | 139 |
| 3.1.6 bsxfun 函数矩阵扩维 | 139 |
| 3.1.7 其他思路 | 145 |
| 3.1.8 扩维思路的总结 | 145 |
| 3.2 数组训练进阶：向量数值为长度的扩维 | 146 |
| 3.2.1 循环 | 147 |
| 3.2.2 利用 arrayfun 扩维 | 148 |
| 3.2.3 利用 repmat 扩维 | 148 |
| 3.2.4 利用 meshgrid 和 ndgrid 扩展矩阵索引 | 149 |
| 3.2.5 利用 bsxfun 扩维 | 150 |
| 3.3 数组训练进阶：求和与构造 | 151 |
| 3.3.1 直接索引法 | 151 |
| 3.3.2 加法中的减法 | 152 |
| 3.3.3 中部元素置零 | 153 |
| 3.3.4 测试矩阵构造 | 153 |
| 3.3.5 卷积和滤波命令 | 157 |
| 3.4 数组训练进阶：“行程长度编码”序列构造 | 160 |
| 3.4.1 利用循环拼接 repmat 扩展矩阵 | 161 |
| 3.4.2 索引扩维、arrayfun 扩展和 cell2mat 拼接 | 161 |
| 3.4.3 按 reshape 变维向量循环处理 | 161 |
| 3.4.4 递归 | 162 |
| 3.4.5 直接调用函数 repelem | 163 |
| 3.5 数组训练进阶：“行程长度编码”的反问题 | 163 |
| 3.5.1 循环拼接向量 | 164 |
| 3.5.2 利用矢量化多次寻址构造序列 | 165 |
| 3.6 数组训练进阶：孤岛测距 | 166 |
| 3.6.1 序列 1, 0 元素索引位相减取最小值 | 166 |
| 3.6.2 直接处理每段“安全”区域 | 167 |
| 3.6.3 利用相邻项数值的构造和比较 | 168 |
| 3.6.4 利用滤波函数 filter2 | 168 |

| | | |
|--------|---|-----|
| 3.7 | 数组训练进阶：生成索引数自扩展序列 | 170 |
| 3.7.1 | 循环拼接 | 171 |
| 3.7.2 | 利用测试矩阵 <code>hankel</code> | 172 |
| 3.7.3 | 利用上三角矩阵函数 <code>triu+meshgrid</code> 构造 | 172 |
| 3.8 | 数组训练进阶：指定子向量长度求均值 | 173 |
| 3.8.1 | 循环逐段求均值 | 174 |
| 3.8.2 | 利用频数累加函数 <code>accumarray</code> | 174 |
| 3.8.3 | 利用测试矩阵 <code>hankel</code> | 176 |
| 3.8.4 | 利用卷积系列命令 | 177 |
| 3.9 | 数组训练进阶：统计群组数量 | 177 |
| 3.9.1 | 循环拼接向量 | 178 |
| 3.9.2 | 涉及排重命令 <code>unique</code> 的几种解法 | 179 |
| 3.9.3 | 利用累积求和函数 <code>cumsum</code> 与 <code>diff</code> | 181 |
| 3.10 | 数组训练进阶：对角矩阵构造 | 181 |
| 3.10.1 | 矩阵叠加 | 182 |
| 3.10.2 | 借助特殊矩阵构造 | 185 |
| 3.10.3 | 循环处理构造思路 | 187 |
| 3.11 | 数组训练进阶：在时间序列中插入 0 元素 | 187 |
| 3.11.1 | 指定位置赋值 | 187 |
| 3.11.2 | 增加 0 元素用 <code>reshape</code> 变维 | 189 |
| 3.11.3 | 循环 | 190 |
| 3.11.4 | 利用 <code>kron</code> 函数扩展矩阵 | 190 |
| 3.11.5 | 正则替换 | 191 |
| 3.12 | 数组训练进阶：Bullseye 矩阵构造 | 191 |
| 3.12.1 | 工具箱特殊函数 | 192 |
| 3.12.2 | 利用特殊矩阵构造 | 194 |
| 3.12.3 | 基本数列构造并矢量化扩维 | 195 |
| 3.12.4 | 递归、判断与循环 | 199 |
| 3.13 | 数组训练进阶：Bullseye 矩阵构造扩展之一 | 200 |
| 3.13.1 | 利用求余命令 <code>mod</code> 或 <code>rem</code> 获得矩阵数值 | 200 |
| 3.13.2 | 利用循环逐元素赋值 | 203 |
| 3.14 | 数组训练进阶：Bullseye 矩阵构造扩展之二 | 204 |
| 3.14.1 | <code>ndgrid</code> 对“基”序列扩维 | 204 |
| 3.14.2 | 利用测试矩阵 <code>spiral</code> 试凑 | 204 |
| 3.15 | 数组训练进阶：Bullseye 矩阵构造扩展之三 | 205 |
| 3.15.1 | 构造“基”序列扩维 | 206 |

| | | |
|--------------|---------------------------|------------|
| 3.15.2 | 特殊矩阵构造 | 209 |
| 3.15.3 | 递归与循环 | 209 |
| 3.16 | 数组训练进阶: Bullseye 矩阵构造扩展之四 | 210 |
| 3.16.1 | 循环 | 211 |
| 3.16.2 | 向量组合 + meshgrid 函数构造 | 212 |
| 3.16.3 | bsxfun 扩维 | 214 |
| 3.16.4 | 测试矩阵 spiral 试凑 | 214 |
| 3.17 | 数组基础训练: 最小值替换为行均值 | 215 |
| 3.17.1 | 循环与矢量化函数二者的结合 | 216 |
| 3.17.2 | 利用高低维索引转换函数 sub2ind | 217 |
| 3.17.3 | 利用稀疏矩阵构造指定位置索引 | 217 |
| 3.17.4 | bsxfun 单一维扩展构造索引 | 217 |
| 3.17.5 | 累积最值函数 cummin | 218 |
| 3.18 | 数组训练进阶: 矩阵元素分隔——“内向”的矩阵 | 219 |
| 3.18.1 | 循环 + 判断 | 220 |
| 3.18.2 | 利用函数 kron 扩维 | 221 |
| 3.18.3 | 利用索引构造变换对新矩阵赋值 | 223 |
| 3.18.4 | 利用稀疏矩阵命令 sparse 构造 | 225 |
| 3.18.5 | 利用累积求和命令 accumarray | 226 |
| 3.19 | 数组训练进阶: 矩阵分块均值——“外向”的矩阵 | 227 |
| 3.19.1 | 循环逐个元素查找相邻索引号 | 227 |
| 3.19.2 | 利用 circshift 函数换序叠加 | 228 |
| 3.19.3 | 利用二维卷积和滤波函数 | 229 |
| 3.20 | 小结 | 229 |
| 第 4 章 | 字符操作进阶: 正则表达式 | 231 |
| 4.1 | 闲话正则 | 231 |
| 4.2 | 灵活的正则语法 | 232 |
| 4.2.1 | 元字符 | 232 |
| 4.2.2 | 转义字符 | 234 |
| 4.2.3 | 匹配次数 | 234 |
| 4.2.4 | 模式 | 236 |
| 4.2.5 | 分组运算 | 237 |
| 4.2.6 | 关于锚点 | 239 |
| 4.2.7 | 左顾右盼 | 239 |
| 4.2.8 | 逻辑与条件运算 | 240 |

| | | |
|--------|-------------------------|-----|
| 4.2.9 | 标记操作 | 241 |
| 4.2.10 | 动态正则表达式 | 243 |
| 4.2.11 | 注释与搜索标识 | 246 |
| 4.3 | 正则表达式基础: 元音字母计数 | 248 |
| 4.3.1 | 其他解法 | 249 |
| 4.3.2 | 正则解法 | 251 |
| 4.4 | 正则表达式基础: 所有的字母都是大写吗? | 252 |
| 4.4.1 | 其他解法 | 252 |
| 4.4.2 | 正则解法 | 254 |
| 4.5 | 正则表达式基础: 移除字符串中的辅音字母 | 255 |
| 4.5.1 | 其他解法 | 255 |
| 4.5.2 | 正则解法 | 258 |
| 4.6 | 正则表达式基础: 首尾元音字母字符串的查找 | 260 |
| 4.6.1 | 其他解法 | 261 |
| 4.6.2 | 正则解法 | 262 |
| 4.7 | 正则表达式基础: 提取文本数字求和 | 263 |
| 4.7.1 | 其他解法 | 263 |
| 4.7.2 | 正则解法 | 265 |
| 4.8 | 正则表达式基础: 钱数统计 | 267 |
| 4.8.1 | 其他解法 | 268 |
| 4.8.2 | 正则解法 | 271 |
| 4.9 | 正则表达式基础: 文本数据的“开关式”查找替换 | 274 |
| 4.9.1 | 其他解法 | 275 |
| 4.9.2 | 正则解法 | 275 |
| 4.10 | 正则表达式基础: 剔除且只剔除首尾指定空格 | 279 |
| 4.10.1 | 其他解法 | 280 |
| 4.10.2 | 正则解法 | 283 |
| 4.11 | 正则表达式基础: 电话区号查询 | 284 |
| 4.11.1 | 其他解法 | 284 |
| 4.11.2 | 正则解法 | 287 |
| 4.12 | 正则表达式基础: 字母出现频数统计 | 288 |
| 4.12.1 | 其他解法 | 289 |
| 4.12.2 | 正则解法 | 292 |
| 4.13 | 正则表达式基础: 翻转单词(不是字母)次序 | 294 |
| 4.13.1 | 其他解法 | 294 |

| | | |
|--------------|-------------------------|------------|
| 4.13.2 | 正则解法 | 296 |
| 4.14 | 正则表达式基础: 寻找最长的“回文”字符 | 298 |
| 4.14.1 | 其他解法 | 298 |
| 4.14.2 | 正则解法 | 299 |
| 4.15 | 正则表达式基础: 求解“字符型”算术题 | 301 |
| 4.15.1 | 其他解法 | 301 |
| 4.15.2 | 正则解法 | 304 |
| 4.16 | 本书前三章中一些问题的正则解法 | 308 |
| 4.16.1 | 正则表达式重解例 1.12 | 308 |
| 4.16.2 | 正则表达式重解例 2.1 | 309 |
| 4.16.3 | 正则表达式重解例 2.5 | 310 |
| 4.16.4 | 正则表达式重解例 2.6 | 310 |
| 4.16.5 | 正则表达式重解例 2.8 | 312 |
| 4.16.6 | 正则表达式重解例 2.9 | 313 |
| 4.16.7 | 正则表达式重解例 2.10 | 314 |
| 4.16.8 | 正则表达式重解例 3.5 | 315 |
| 4.16.9 | 正则表达式重解例 3.6 | 315 |
| 4.16.10 | 正则表达式重解例 3.7 | 319 |
| 4.17 | 小 结 | 319 |
| 第 5 章 | 多维数组漫谈 | 320 |
| 5.1 | 多维数组基础 | 321 |
| 5.2 | 多维数组问题 1: 扩维 | 328 |
| 5.2.1 | 利用 kron 和 reshape 函数 | 330 |
| 5.2.2 | 利用 cat 函数 | 332 |
| 5.2.3 | 利用 bsxfun 和 shiftdim 函数 | 337 |
| 5.2.4 | 利用 convn 和 shiftdim 函数 | 340 |
| 5.3 | 多维数组问题 2: “乘”操作 | 340 |
| 5.3.1 | 循环和分情况判断的基本方法 | 341 |
| 5.3.2 | 点积单独构造维数向量与循环的组合 | 343 |
| 5.3.3 | 利用高、低维索引变换 | 343 |
| 5.3.4 | cell 数组结构与 repmat 函数组合 | 346 |
| 5.3.5 | cell 数组结构 + 扩维 | 349 |
| 5.4 | 多维数组问题 3: 高维数组的矢量化索引寻址 | 352 |
| 5.4.1 | permute 做源数据维度变换的不同方式 | 354 |
| 5.4.2 | 索引分组 | 360 |

| | |
|------------------------------------|------------|
| 5.5 小 结 | 361 |
| 第 6 章 匿名函数专题 | 362 |
| 6.1 匿名函数探析 | 362 |
| 6.1.1 基本应用 | 362 |
| 6.1.2 匿名函数嵌套构造函数在程序编写中的应用 | 364 |
| 6.1.3 匿名函数与参数传递 | 367 |
| 6.1.4 匿名函数进阶 | 376 |
| 6.2 匿名函数应用：函数迭代器 | 381 |
| 6.2.1 循环求解的多个变体 | 381 |
| 6.2.2 递归思路及引申 | 382 |
| 6.3 匿名函数应用：返回多输出 | 385 |
| 6.3.1 利用匿名函数创建多输出句柄 | 385 |
| 6.3.2 利用匿名函数构造更灵活的任意数量输出 | 390 |
| 6.4 匿名函数应用：复合句柄 | 393 |
| 6.4.1 利用子函数 | 394 |
| 6.4.2 利用匿名函数构造 | 395 |
| 6.5 匿名函数应用：斐波那契数列求值 | 400 |
| 6.5.1 几种不用匿名函数定义句柄的解法 | 401 |
| 6.5.2 使用匿名函数构造序列的相关算法 | 404 |
| 6.6 匿名函数应用：斐波那契数列构造 | 406 |
| 6.6.1 不使用匿名函数的几种求解思路 | 407 |
| 6.6.2 使用匿名函数构造受控句柄的几种解法 | 409 |
| 6.7 匿名函数应用：函数执行计数器中的匿名函数传参机理 | 410 |
| 6.7.1 save+load 存储调用变量 | 412 |
| 6.7.2 图形句柄 | 413 |
| 6.7.3 随机数控制器 rng | 414 |
| 6.7.4 全局变量定义“global” | 416 |
| 6.7.5 匿名函数句柄传递计数结果 | 417 |
| 6.8 小 结 | 423 |
| 参考文献 | 424 |