面向深度学习的 R 语言线性代数速查

赵奇

2015 年 *8* 月 *17* 日

Contents

```
library(knitr)
#library(printr)
library(showtext)
```

```
## Loading required package: sysfonts
```

```
opts_chunk$set(fig.showtext = TRUE,prompt = TRUE,message = FALSE,warning = FALSE,cache = TRUE)
```

1 矩阵乘法

1.1 乘法

```
> A <- matrix(data = 1:36, nrow = 6)
> A
```

1	7	13	19	25	31
2	8	14	20	26	32
3	9	15	21	27	33
4	10	16	22	28	34
5	11	17	23	29	35
6	12	18	24	30	36

```
> B <- matrix(data = 1:30, nrow = 6)
> B
```

1	7	13	19	25
2	8	14	20	26
3	9	15	21	27
4	10	16	22	28
5	11	17	23	29
6	12	18	24	30

```
> A %*% B
```

441	1017	1593	2169	2745
462	1074	1686	2298	2910
483	1131	1779	2427	3075
504	1188	1872	2556	3240
525	1245	1965	2685	3405

546	1302	2058	2814	3570

1.2 Hadamard 乘法

```
> A <- matrix(data = 1:36, nrow = 6)
> A
```

1	7	13	19	25	31
2	8	14	20	26	32
3	9	15	21	27	33
4	10	16	22	28	34
5	11	17	23	29	35
6	12	18	24	30	36

```
> B <- matrix(data = 11:46, nrow = 6)
> B
```

11	17	23	29	35	41
12	18	24	30	36	42
13	19	25	31	37	43
14	20	26	32	38	44
15	21	27	33	39	45
16	22	28	34	40	46

```
> A * B
```

11	119	299	551	875	1271
24	144	336	600	936	1344
39	171	375	651	999	1419
56	200	416	704	1064	1496
75	231	459	759	1131	1575
96	264	504	816	1200	1656

1.3 点乘

```
> X <- matrix(data = 1:10, nrow = 10)
> X
```

```
> Y <- matrix(data = 11:20, nrow = 10)
> Y
```

```
> dotProduct <- function(X, Y) {
+     as.vector(t(X) %*% Y)
+ }
>
> dotProduct(X, Y)
```

```
## [1] 935
```

1.4 矩阵相乘的性质

1.4.1 满足分配率

```
> A <- matrix(data = 1:25, nrow = 5)
> B <- matrix(data = 26:50, nrow = 5)
> C <- matrix(data = 51:75, nrow = 5)
>
> A %*% (B + C)
```

4555	5105	5655	6205	6755
4960	5560	6160	6760	7360
5365	6015	6665	7315	7965
5770	6470	7170	7870	8570
6175	6925	7675	8425	9175

```
> A %*% B + A %*% C
```

4555	5105	5655	6205	6755
4960	5560	6160	6760	7360
5365	6015	6665	7315	7965
5770	6470	7170	7870	8570
6175	6925	7675	8425	9175

1.4.2 满足结合律

```
> A <- matrix(data = 1:25, nrow = 5)
> B <- matrix(data = 26:50, nrow = 5)
> C <- matrix(data = 51:75, nrow = 5)
>
> (A %*% B) %*% C
```

569850	623350	676850	730350	783850
620450	678700	736950	795200	853450
671050	734050	797050	860050	923050
721650	789400	857150	924900	992650
772250	844750	917250	989750	1062250

```
> A %*% (B %*% C)
```

569850	623350	676850	730350	783850
620450	678700	736950	795200	853450
671050	734050	797050	860050	923050
721650	789400	857150	924900	992650
772250	844750	917250	989750	1062250

1.4.3 不满足交换律

```
> A <- matrix(data = 1:25, nrow = 5)
> B <- matrix(data = 26:50, nrow = 5)
>
> A %*% B
```

1590	1865	2140	2415	2690
1730	2030	2330	2630	2930
1870	2195	2520	2845	3170
2010	2360	2710	3060	3410
2150	2525	2900	3275	3650

```
> B %*% A
```

590	1490	2390	3290	4190
605	1530	2455	3380	4305
620	1570	2520	3470	4420
635	1610	2585	3560	4535
650	1650	2650	3650	4650

2 矩阵转置

```
> A <- matrix(data = 1:25, nrow = 5, ncol = 5, byrow = TRUE)
> A
```

1	2	3	4	5
6	7	8	9	10
11	12	13	14	15
16	17	18	19	20
21	22	23	24	25

```
> t(A)
```

1	6	11	16	21
2	7	12	17	22
3	8	13	18	23
4	9	14	19	24
5	10	15	20	25

2.1 矩阵转置的性质

```
> A <- matrix(data = 1:25, nrow = 5)
> B <- matrix(data = 25:49, nrow = 5)
>
```

```
> t(A %*% B)
```

1535	1670	1805	1940	2075
1810	1970	2130	2290	2450
2085	2270	2455	2640	2825
2360	2570	2780	2990	3200
2635	2870	3105	3340	3575

```
> t(B) %*% t(A)
```

1535	1670	1805	1940	2075
1810	1970	2130	2290	2450
2085	2270	2455	2640	2825
2360	2570	2780	2990	3200
2635	2870	3105	3340	3575

3 解线性方程 $\mathbf{Ax} = \mathbf{B}$

3.1 方法 1

```
> A <- matrix(data = c(1, 3, 2, 4, 2, 4, 3, 5, 1, 6, 7, 2, 1, 5, 6, 7), nrow = 4, byrow = TRUE)
> A
```

1	3	2	4
2	4	3	5
1	6	7	2
1	5	6	7

```
> B <- matrix(data = c(1, 2, 3, 4), nrow = 4)
> B
```

```
> solve(a = A, b = B)
```

0.6153846
-0.8461538
1.0000000
0.2307692

3.2 方法 2

```
> A <- matrix(data = c(1, 3, 2, 4, 2, 4, 3, 5, 1, 6, 7, 2, 1, 5, 6, 7), nrow = 4, byrow = TRUE)
> A
```

1	3	2	4
2	4	3	5
1	6	7	2
1	5	6	7

```
> B <- matrix(data = c(1, 2, 3, 4), nrow = 4)
> B
```

```
> library(MASS)
>
> X <- ginv(A) %*% B
> X
```

0.6153846
-0.8461538
1.0000000
0.2307692

4 单位阵

```
> I <- diag(x = 1, nrow = 5, ncol = 5)
> I
```

1	0	0	0	0
0	1	0	0	0
0	0	1	0	0
0	0	0	1	0
0	0	0	0	1

```
> A <- matrix(data = 1:25, nrow = 5)
>
> A %*% I
```

1	6	11	16	21
2	7	12	17	22
3	8	13	18	23
4	9	14	19	24
5	10	15	20	25

```
> I %*% A
```

1	6	11	16	21
2	7	12	17	22
3	8	13	18	23
4	9	14	19	24

5	10	15	20	25

5 矩阵求逆

```
> A <- matrix(data = c(1, 2, 3, 1, 2, 3, 4, 5, 6, 2, 3, 4, 5, 6, 7, 8, 9, 1, 2, 3, 4, 5, 6, 7, 3), nrow
> A
```

1	3	3	8	4
2	4	4	9	5
3	5	5	1	6
1	6	6	2	7
2	2	7	3	3

```
> library(MASS)
>
> ginv(A)
```

-0.3333333	0.3333333	0.3333333	-0.3333333	0.0
-4.0888889	3.6444444	-1.2222222	0.8666667	-0.2
-0.3555556	0.2444444	-0.2222222	0.1333333	0.2
-0.1111111	0.2222222	-0.1111111	0.0000000	0.0
3.8888889	-3.4444444	1.2222222	-0.6666667	0.0

```
> ginv(A) %*% A
```

1	0	0	0	0
0	1	0	0	0
0	0	1	0	0
0	0	0	1	0
0	0	0	0	1

```
> A %*% ginv(A)
```

1	0	0	0	0
0	1	0	0	0
0	0	1	0	0
0	0	0	1	0
0	0	0	0	1

6 行列式

```
> A <- matrix(data = c(1, 3, 2, 4, 2, 4, 3, 5, 1, 6, 7, 2, 1, 5, 6, 7), nrow = 4, byrow = TRUE)
> A
```

1	3	2	4
2	4	3	5
1	6	7	2
1	5	6	7

```
> det(A)
```

```
## [1] -39
```

7 范数

```
> lpNorm <- function(A, p) {
+     if (p >= 1 & dim(A)[[2]] == 1 && is.infinite(p) == FALSE) {
+             sum((apply(X = A, MARGIN = 1, FUN = abs)) ** p) ** (1 / p)
+
+     } else if (p >= 1 & dim(A)[[2]] == 1 & is.infinite(p)) {
+         max(apply(X = A, MARGIN = 1, FUN = abs)) # Max Norm
+     } else {
+         invisible(NULL)
+     }
+ }
>
> lpNorm(A = matrix(data = 1:10), p = 1)
```

```
## [1] 55
```

```
> lpNorm(A = matrix(data = 1:10), p = 2) # Euclidean Distance
```

```
## [1] 19.62142
```

```
> lpNorm(A = matrix(data = 1:10), p = 3)
```

```
## [1] 14.46245
```

```
> lpNorm(A = matrix(data = -100:10), p = Inf)
```

```
## [1] 100
```

7.1 性质

```
> lpNorm(A = matrix(data = rep(0, 10)), p = 1) == 0
```

```
## [1] TRUE
```

```
> lpNorm(A = matrix(data = 1:10) + matrix(data = 11:20), p = 1) <= lpNorm(A = matrix(data = 1:10), p =
```

```
## [1] TRUE
```

```
> tempFunc <- function(i) {
+     lpNorm(A = i * matrix(data = 1:10), p = 1) == abs(i) * lpNorm(A = matrix(data = 1:10), p = 1)
+ }
>
> all(sapply(X = -10:10, FUN = tempFunc))
```

```
## [1] TRUE
```

8 Frobenius 范数

```
> frobeniusNorm <- function(A) {
+     (sum((as.numeric(A)) ** 2)) ** (1 / 2)
+ }
>
> frobeniusNorm(A = matrix(data = 1:25, nrow = 5))
```

```
## [1] 74.33034
```

9 特殊矩阵和向量

9.1 对角矩阵

```
> A <- diag(x = c(1:5, 6, 1, 2, 3, 4), nrow = 10)
> A
```

1	0	0	0	0	0	0	0	0	0
0	2	0	0	0	0	0	0	0	0
0	0	3	0	0	0	0	0	0	0
0	0	0	4	0	0	0	0	0	0
0	0	0	0	5	0	0	0	0	0
0	0	0	0	0	6	0	0	0	0
0	0	0	0	0	0	1	0	0	0
0	0	0	0	0	0	0	2	0	0
0	0	0	0	0	0	0	0	3	0
0	0	0	0	0	0	0	0	0	4

```
> X <- matrix(data = 21:30)
> X
```

```
> A %*% X
```

21
44
69
96
125
156
27
56
87
120

```r
> library(MASS)
>
> ginv(A)
```

1	0.0	0.0000000	0.00	0.0	0.0000000	0	0.0	0.0000000	0.00
0	0.5	0.0000000	0.00	0.0	0.0000000	0	0.0	0.0000000	0.00
0	0.0	0.3333333	0.00	0.0	0.0000000	0	0.0	0.0000000	0.00
0	0.0	0.0000000	0.25	0.0	0.0000000	0	0.0	0.0000000	0.00
0	0.0	0.0000000	0.00	0.2	0.0000000	0	0.0	0.0000000	0.00
0	0.0	0.0000000	0.00	0.0	0.1666667	0	0.0	0.0000000	0.00
0	0.0	0.0000000	0.00	0.0	0.0000000	1	0.0	0.0000000	0.00
0	0.0	0.0000000	0.00	0.0	0.0000000	0	0.5	0.0000000	0.00
0	0.0	0.0000000	0.00	0.0	0.0000000	0	0.0	0.3333333	0.00
0	0.0	0.0000000	0.00	0.0	0.0000000	0	0.0	0.0000000	0.25

9.2 对称矩阵

```r
> A <- matrix(data = c(1, 2, 2, 1), nrow = 2)
> A
```

1	2
2	1

```r
> all(A == t(A))
```

```
## [1] TRUE
```

9.3 单位向量

```r
> lpNorm(A = matrix(data = c(1, 0, 0, 0)), p = 2)
```

```
## [1] 1
```

9.4 正交向量

```
> X <- matrix(data = c(11, 0, 0, 0))
> Y <- matrix(data = c(0, 11, 0, 0))
>
> all(t(X) %*% Y == 0)
```

```
## [1] TRUE
```

9.5 正交单位向量组

```
> X <- matrix(data = c(1, 0, 0, 0))
> Y <- matrix(data = c(0, 1, 0, 0))
>
> lpNorm(A = X, p = 2) == 1
```

```
## [1] TRUE
```

```
> lpNorm(A = Y, p = 2) == 1
```

```
## [1] TRUE
```

```
> all(t(X) %*% Y == 0)
```

```
## [1] TRUE
```

9.6 正交矩阵

```
> A <- matrix(data = c(1, 0, 0, 0, 1, 0, 0, 0, 1), nrow = 3, byrow = TRUE)
> A
```

1	0	0
0	1	0
0	0	1

```
> all(t(A) %*% A == A %*% t(A))
```

```
## [1] TRUE
```

```
> all(t(A) %*% A == diag(x = 1, nrow = 3))
```

```
## [1] TRUE
```

```
> library(MASS)
> all(t(A) == ginv(A))
```

```
## [1] TRUE
```

9.7 特征分解

```
> A <- matrix(data = 1:25, nrow = 5, byrow = TRUE)
> A
```

1	2	3	4	5
6	7	8	9	10
11	12	13	14	15
16	17	18	19	20
21	22	23	24	25

```
> y <- eigen(x = A)
> y
```

```
## $values
## [1]  6.864208e+01 -3.642081e+00  4.626054e-15  7.173861e-16 -1.202776e-16
##
## $vectors
##            [,1]        [,2]        [,3]        [,4]        [,5]
## [1,] -0.1079750 -0.67495283  0.43684670 -0.10763824 -0.10839336
## [2,] -0.2527750 -0.36038970 -0.80161272 -0.29282210  0.05352876
## [3,] -0.3975750 -0.04582657  0.38296630  0.85748010  0.52747309
## [4,] -0.5423751  0.26873656 -0.10848123 -0.40594096 -0.78195903
## [5,] -0.6871751  0.58329969  0.09028096 -0.05107881  0.30935054
```

```
> library(MASS)
> all.equal(y$vectors %*% diag(y$values) %*% ginv(y$vectors), A)
```

```
## [1] TRUE
```

9.8 奇异值分解

```
> A <- matrix(data = 1:36, nrow = 6, byrow = TRUE)
> A
```

1	2	3	4	5	6
7	8	9	10	11	12
13	14	15	16	17	18
19	20	21	22	23	24
25	26	27	28	29	30
31	32	33	34	35	36

```
> y <- svd(x = A)
> y
```

```
## $d
## [1] 1.272064e+02 4.952580e+00 3.378635e-15 8.394570e-16 1.033790e-16
## [6] 3.580178e-17
```

```
##
## $u
##             [,1]        [,2]       [,3]        [,4]        [,5]
## [1,] -0.06954892 -0.72039744  0.6825128 -0.09839656  0.02608853
## [2,] -0.18479698 -0.51096788 -0.5968445 -0.37321714 -0.37506104
## [3,] -0.30004504 -0.30153832 -0.2663298  0.69016113  0.46300174
## [4,] -0.41529310 -0.09210875 -0.1904227 -0.24789729  0.33483806
## [5,] -0.53054116  0.11732081  0.1546485  0.41016255 -0.68887985
## [6,] -0.64578922  0.32675037  0.2164357 -0.38081269  0.24001255
##              [,6]
## [1,] -0.002050105
## [2,]  0.261871683
## [3,]  0.239631064
## [4,] -0.780523688
## [5,] -0.195082027
## [6,]  0.476153071
##
## $v
##            [,1]        [,2]       [,3]       [,4]        [,5]         [,6]
## [1,] -0.3650545  0.62493577  0.3344540  0.5910570 -0.11967487  0.025700735
## [2,] -0.3819249  0.38648609 -0.5111072 -0.2274676  0.58187258 -0.230681802
## [3,] -0.3987952  0.14803642 -0.2094366 -0.4026879 -0.44648506  0.643177640
## [4,] -0.4156655 -0.09041326  0.1962240 -0.3319529 -0.44312743 -0.688468671
## [5,] -0.4325358 -0.32886294  0.6080205 -0.1734453  0.49659403  0.241627956
## [6,] -0.4494062 -0.56731262 -0.4181548  0.5444966 -0.06917926  0.008644142
```

```
> all.equal(y$u %*% diag(y$d) %*% t(y$v), A)
```

```
## [1] TRUE
```

9.9 广义逆矩阵

```
> A <- matrix(data = 1:25, nrow = 5)
> A
```

1	6	11	16	21
2	7	12	17	22
3	8	13	18	23
4	9	14	19	24
5	10	15	20	25

```
> B <- ginv(A)
> B
```

-0.152	-0.08	-0.008	0.064	0.136
-0.096	-0.05	-0.004	0.042	0.088
-0.040	-0.02	0.000	0.020	0.040
0.016	0.01	0.004	-0.002	-0.008
0.072	0.04	0.008	-0.024	-0.056

```r
> y <- svd(A)
> all.equal(y$v %*% ginv(diag(y$d)) %*% t(y$u), B)
```

```
## [1] TRUE
```

9.10 矩阵的迹

```r
> A <- diag(x = 1:10)
> A
```

1	0	0	0	0	0	0	0	0	0
0	2	0	0	0	0	0	0	0	0
0	0	3	0	0	0	0	0	0	0
0	0	0	4	0	0	0	0	0	0
0	0	0	0	5	0	0	0	0	0
0	0	0	0	0	6	0	0	0	0
0	0	0	0	0	0	7	0	0	0
0	0	0	0	0	0	0	8	0	0
0	0	0	0	0	0	0	0	9	0
0	0	0	0	0	0	0	0	0	10

```r
> library(psych)
> tr(A)
```

```
## [1] 55
```

```r
> alternativeFrobeniusNorm <- function(A) {
+     sqrt(tr(t(A) %*% A))
+ }
>
> alternativeFrobeniusNorm(A)
```

```
## [1] 19.62142
```

```r
> frobeniusNorm(A)
```

```
## [1] 19.62142
```

```r
> all.equal(tr(A), tr(t(A)))
```

```
## [1] TRUE
```

```r
> A <- diag(x = 1:5)
> A
```

1	0	0	0	0
0	2	0	0	0
0	0	3	0	0
0	0	0	4	0
0	0	0	0	5

```r
> B <- diag(x = 6:10)
> B
```

6	0	0	0	0
0	7	0	0	0
0	0	8	0	0
0	0	0	9	0
0	0	0	0	10

```r
> C <- diag(x = 11:15)
> C
```

11	0	0	0	0
0	12	0	0	0
0	0	13	0	0
0	0	0	14	0
0	0	0	0	15

```r
> all.equal(tr(A %*% B %*% C), tr(C %*% A %*% B))
```

```
## [1] TRUE
```

```r
> all.equal(tr(C %*% A %*% B), tr(B %*% C %*% A))
```

```
## [1] TRUE
```