Carson Slater

6376 Homework 7 Solutions

```
source("../helpers.R")
```

Preliminary Citations

In question 3, for my `emax()` power iteration function, I checked my method for generating a vector on the unit sphere with ChatGPT 3.5. I consulted Trefethen and Bau[1] for a more intuitive understand for all functions we were using. All functions such as `positize()` or including `positize()` were derived from the homework assignment for this course.

Question 1

(a)

Suppose $\mathbf{a}$, $\mathbf{b} \in \mathbb{C}^m$, where $\mathbf{a} \neq 0_m$. We can suggest that some scalar $y = \mathbf{a}x$, where $\text{Proj}_{\mathbf{a}}\mathbf{b} = y \in \mathbb{C}^m$ and $x \in \mathbb{R}$, since the projection of $\mathbf{b}$ onto $\mathbf{a}$ is a multiple of $\mathbf{a}$. Since the projection is orthogonal, as well as the fact that $\mathbf{a}$ and $\mathbf{a} - \mathbf{b}x$ are orthogonal, we can say their inner product is zero $\mathbf{a}'(\mathbf{a} - \mathbf{b}x) = 0$. From this result we get $\mathbf{a}'\mathbf{a} = \mathbf{a}'\mathbf{b}x$, which yields $x = \frac{\mathbf{a}'\mathbf{b}}{\mathbf{a}'\mathbf{a}}$. So then the projection of $\mathbf{b}$ onto $\mathbf{a}$ would be $y = \text{Proj}_{\mathbf{a}}\mathbf{b} = \frac{\mathbf{a}'\mathbf{b}}{\mathbf{a}'\mathbf{a}}\mathbf{a}$.

(b)

If the euclidean norm of $\mathbf{a} = 1$, the denominator of the equation $\text{Proj}_{\mathbf{a}}\mathbf{b}$ becomes 1 because the inner product of $\mathbf{a}'\mathbf{a} = 1$ when $||\mathbf{a}||_2 = ||\mathbf{q}||_2 = 1$. So the formula becomes $\text{Proj}_{\mathbf{q}}\mathbf{b} = \mathbf{q}'\mathbf{b}\mathbf{q}$.

(c)

Suppose n linearly independent column vectors $\mathbf{a}_1, \mathbf{a}_2, ..., \mathbf{a}_n \in \mathbb{C}^m$ comprise matrix $\mathbf{A} \in \mathbb{C}^{m \times n}$ such that $m > n$, and that $\mathbf{b} \in \mathbb{R}^m$. We know that the projection of $\mathbf{b}$ onto each column of $\mathbf{A}$ is a linear combination of $\mathbf{A}$, and hence can be expressed using some $\mathbf{x} \in \mathbb{R}^n$ as $\text{Proj}_{\mathbf{A}}\mathbf{b} = \mathbf{A}\mathbf{x}$. Additionally, since it is an orthogonal projection, the vector $\mathbf{b} - \mathbf{A}\mathbf{x}$ must be orthogonal to each $\mathbf{a}_i$ in $\mathbf{A}$; hence $\mathbf{a}_i'(\mathbf{b} - \mathbf{A}\mathbf{x}) = 0$ for each i^{th} column vector in $\mathbf{A}$, which is equivalent to $\mathbf{A}'(\mathbf{b} - \mathbf{A}\mathbf{x}) = 0$. Since each $\mathbf{a}_i$ is linearly independent, we know the matrix is full rank,

[1] Trefethen, Lloyd Nicholas, and David Bau. *Numerical Linear Algebra*. United States: Society for Industrial and Applied Mathematics, 1997.

which implies $\mathbf{A}'\mathbf{A}$ is invertible. Since $\mathbf{A}'\mathbf{A}$ is invertible, then $(\mathbf{A}'\mathbf{A})^{-1}$ exists, and we can solve $\mathbf{A}'(\mathbf{b} - \mathbf{A}\mathbf{x}) = 0$ for $\mathbf{x}$. Therefore $\mathbf{x} = (\mathbf{A}'\mathbf{A})^{-1}\mathbf{A}'\mathbf{b}$. So then we have that $\text{Proj}_{\mathbf{A}}\mathbf{b} = \mathbf{A}(\mathbf{A}'\mathbf{A})^{-1}\mathbf{A}'\mathbf{b}$.

(d)

If each $\mathbf{a}_i$ is orthonormal, then $\mathbf{A}$ is orthogonal, which we can rewrite as $\mathbf{Q}$. Since the product of orthogonal matrices with their conjugate transposes is the $\mathbf{I}_m$, we have that $\text{Proj}_{\mathbf{Q}}\mathbf{b} = \mathbf{Q}(\mathbf{Q}'\mathbf{Q})^{-1}\mathbf{Q}'\mathbf{b} = \mathbf{Q}(\mathbf{I}_m)^{-1}\mathbf{Q}'\mathbf{b} = \mathbf{Q}\mathbf{Q}'\mathbf{b}$.

Question 2

I have read/perused the appropriate specified Trefethen and Bau lectures (24-28).

Question 3

```r
# using positize function from the homework specifications
positize <- function(x) sign(x[1]) * x

# emax()
emax <- function(A, tol = sqrt(.Machine$double.eps)) {

  # generate a random number on the unit sphere
  v <- rnorm(nrow(A)) |>
    normalize()

  # iterations
  while(TRUE) {
    new_vec <- mat_vec_prod(A, v) |>
      normalize() |>
      positize()
    if (all((new_vec - v) < tol)) break
    v <- new_vec
  }

  # output
  list("value" = mat_vec_prod(A, t(new_vec)) |> dot(new_vec),
       "vector" = new_vec)
}
```

```r
(A <- matrix(c(1:4, 6:10), nrow = 3, byrow = TRUE))
#      [,1] [,2] [,3]
# [1,]    1    2    3
# [2,]    4    6    7
# [3,]    8    9   10

# testing emax()
emax(A)
# $value
# [1] 18.03215
#
# $vector
# [1] 0.2071014 0.5424913 0.8141328

eigen(A)
# eigen() decomposition
# $values
# [1] 18.0321458 -1.3251013  0.2929554
#
# $vectors
#            [,1]       [,2]       [,3]
# [1,] -0.2071014 -0.6238002  0.3599805
# [2,] -0.5424913 -0.3344309 -0.8132015
# [3,] -0.8141328  0.7064201  0.4572935

emax(-A)
# $value
# [1] -18.03215
#
# $vector
# [1] 0.2071014 0.5424913 0.8141328
```

Question 4

```r
# using positize function from the homework specifications
positize_first_row <- function(A) apply(A, 2, positize)

# eigen_qr()
eigen_qr <- function(A, tol = sqrt(.Machine$double.eps)){
```

```
  # A, per the homework much be m x m
  stopifnot(nrow(A) == ncol(A))

  # initialize start A
  Ak_1 <- A

  # Start with U = I_n
  U <- diag(rep(1, nrow(A)))
  while(TRUE) {
    QR <- QR(Ak_1)

    Ak <- mat_mat_prod(QR$R, QR$Q)

    U <- mat_mat_prod(U, QR$Q) |>
      positize_first_row()

    # stopping criterion requires keeping track of A_k and A_{k-1}
    if (mat_norm(Ak - Ak_1, type = "f") < tol) break

    Ak_1 <- Ak
  }
  # returning values
  list("value" = Ak |> diag(),
       "vector" = U)
}
```

```
(A <- matrix(c(1:4, 6:10), nrow = 3, byrow = TRUE))
#      [,1] [,2] [,3]
# [1,]    1    2    3
# [2,]    4    6    7
# [3,]    8    9   10

(A <- crossprod(A))
#      [,1] [,2] [,3]
# [1,]   81   98  111
# [2,]   98  121  138
# [3,]  111  138  158

# testing eigen_QR()
eigen_qr(A)
# $value
```

```
# [1] 357.91047441   2.02181118   0.06771441
#
# $vector
#           [,1]       [,2]       [,3]
# [1,] 0.4883746  0.8442564  0.2207293
# [2,] 0.5795159 -0.1246686 -0.8053689
# [3,] 0.6524198 -0.5212379  0.5501449

eigen(A)
# eigen() decomposition
# $values
# [1] 357.91047441   2.02181118   0.06771441
#
# $vectors
#            [,1]       [,2]       [,3]
# [1,] -0.4715411  0.8359628  0.2807404
# [2,] -0.5813187 -0.0552796 -0.8117960
# [3,] -0.6631120 -0.5459948  0.5120274
```

Question 5

```
det_qr <- function(A) {
  eigen_qr(A)$value |>
    log() |>
    sum() |>
    exp()
}
```

```
set.seed(2)
(A <- matrix(rpois(64, lambda = 5), nrow = 8))
#      [,1] [,2] [,3] [,4] [,5] [,6] [,7] [,8]
# [1,]    3    5   10    4    7   10    5    7
# [2,]    6    5    3    5    8    4    7   11
# [3,]    5    5    5    3    5    2    1    5
# [4,]    3    3    2    4    6    3    1    6
# [5,]    9    6    6    9    7    9    6    7
# [6,]    9    3    4    3    4    7    8    8
# [7,]    3    4    7    1    6   10    4    6
# [8,]    7    7    3    3    3    4    7    3
```

```r
(A <- crossprod(A))
#      [,1] [,2] [,3] [,4] [,5] [,6] [,7] [,8]
# [1,]  299  221  211  201  250  275  252  304
# [2,]  221  194  193  160  217  232  193  244
# [3,]  211  193  248  160  240  292  195  265
# [4,]  201  160  160  166  197  202  165  224
# [5,]  250  217  240  197  284  293  221  324
# [6,]  275  232  292  202  293  375  261  333
# [7,]  252  193  195  165  221  261  241  274
# [8,]  304  244  265  224  324  333  274  389

det_qr(A)
# [1] 220047556

det(A)
# [1] 220047556
```