Homework 1

Jimmy Hickey

Due @ 5pm on January 24, 2020

Part 1

1

Let $\mathbf{v} \in \mathbb{R}^n$. What is the computational complexity of solving the linear system $\mathbf{Ax} = \mathbf{b}$ where $\mathbf{A} = \mathbf{I} + \mathbf{v}\mathbf{v}^\mathsf{T}$. Include the cost of constructing $\mathbf{A}$ explicitly.

Answer:

Let's start by finding A. To do the vector multiplication vv^T will take $\mathcal{O}(n^2)$. Then adding this to the identity will be $\mathcal{O}(n)$. So overall constructing A will be $\mathcal{O}(n^2)$.

$$A = \overbrace{I + \underbrace{vv^T}_{n^2}}^{n} \Rightarrow \mathcal{O}(n^2 + n) = \mathcal{O}(n^2)$$

To solve the equation $Ax = b$, we can make transform A into an upper triangular matrix (performing the same operations on b). Lets look at making the first column all 0's except the first entry. To make the first column of the second row ($a_{1,2}$) zero, we need to multiple row 1 by a scalar ($\mathcal{O}(n)$) and then subtract that from row 2 ($\mathcal{O}(n)$). This will have to be repeated for each for ($n-1$ times). This gives $\mathcal{O}((n-1)(n+n)) = \mathcal{O}(n^2)$.

$$\underbrace{(n-1)(\overbrace{a \cdot \text{row } 1}^{\text{scalar multiplication: } \mathcal{O}(n)} \quad \overbrace{+\text{row } i}^{\text{vector addition: } \mathcal{O}(n)})}_{\text{repeat for all rows: } \mathcal{O}(n)} = \mathcal{O}(n(n+n)) = \mathcal{O}(n^2)$$

We follow similar steps for row 2, except now we only have to repeat is $n-2$ times.

$$\underbrace{(n-2)(\overbrace{a \cdot \text{row } 2}^{\text{scalar multiplication: } \mathcal{O}(n)} \quad \overbrace{+\text{row } i}^{\text{vector addition: } \mathcal{O}(n)})}_{\text{repeat for all rows: } \mathcal{O}(n)} = \mathcal{O}(n(n+n)) = \mathcal{O}(n^2)$$

We repeat this process until we get to the last row which will be all 0's except for the nth entry; it will be repeated $n-1$ times.

$$(n-1)\begin{cases} \mathcal{O}(n^2) \\ \mathcal{O}(n^2) \\ \dots \\ \mathcal{O}(n^2) \end{cases}$$

Thus, the whole process has $\mathcal{O}((n-1)n^2) = \mathcal{O}(n^3)$ complexity.

2

2. Use the Sherman-Morrison-Woodbury formula (sometimes called the Matrix Inversion Lemma) to solve the linear system in question 1 more efficiently.

Answer: One application of the Sherman-Morrison-Woodbury formula is

$$(A + uv^T)^{-1} = A^{-1} - \frac{A^{-1}uv^TA^{-1}}{1 + v^TA^-1u}.$$

At first this may look like it increases complexity, but if A^{-1} is already known, it saves an expensive inversion process. In our case,

$$Ax = b \to x = A^{-1}b = \Big[I - vv^T\Big]^{-1} b = \Big[I^{-1} - \frac{I^{-1}vv^TI^{-1}}{1 + v^TI^{-1}v}\Big]b.$$

Since we know $I = I^{-1}$, this simplifies to

$$x = \Big[I - \frac{Ivv^TI}{1 + v^TIv}\Big]b.$$

Let's look at the pieces of this equation individually. Starting with the numerator.

From left to right, v^TI is $\mathcal{O}(n)$ since we only need to multiply/replace the diagonals. Then $v(v^TI)$ is $\mathcal{O}(n)$ since v^TI only has nonzero entries on the diagonals, so there are only n multiplcations. Finally, $I(v(v^TI))$ is again $\mathcal{O}(n)$ for the same reason as before.

$$\underbrace{I\,\overbrace{v\,\underbrace{v^TI}_{\mathcal{O}(n)}}^{\mathcal{O}(n)}}_{\mathcal{O}(n)} = \mathcal{O}(n + n + n) = \mathcal{O}(n)$$

We can use the same logic to look at the complexity of the denominator.

$$\overbrace{1 + \underbrace{v^T\ \underbrace{Iv}_{\mathcal{O}(n)}}_{\mathcal{O}(n)}}^{1} = \mathcal{O}(n + n + n) = \mathcal{O}(n)$$

The division between these is $\mathcal{O}(n)$ since we only need to divide the diagonals (since all of the off diagonals are 0). Similarly, $I - \frac{Ivv^TI}{1+v^TIv}$ again, only acts on the diagonals which is also $\mathcal{O}(n)$. Finally, the matrix-vector multiplication of $(I - \frac{Ivv^TI}{1+v^TIv})b$ is $\mathcal{O}(n^2)$. This gives us $\mathcal{O}(n^2)$ overall.

3

3. Let $f : \mathbb{R}^n \mapsto \mathbb{R}$ that is differentiable and convex. Let X be an n-dimensional random vector whose first moment exists. Prove that

$$f(\mathbb{E}[X]) \leq \mathbb{E}[f(X)].$$

Note that it is not necessary to assume that f is differentiable. Hint: use the fact that the best first order approximation to a convex function is a global underestimator of it.

Answer:

Using a the first order approximation as a global underestimator, we get

$$f(X) \geq f(E(X)) + \nabla(f(E(X)))^T(X - E(X))$$

Now we will take the expectation of both sides.

$$\begin{aligned} E\Big[f(X)\Big] &\geq E\Big[f(E(X)) + \nabla(f(E(X)))^T(X - E(X))\Big] \\ &= E\Big[f(E(X))\Big] + E\Big[\nabla(f(E(X)))^T(X - E(X))\Big] \\ &= f(E(X)) + \nabla(f(E(X)))^T \cdot E\Big[(X - E(X))\Big] && \text{constants} \\ &= f(E(X)) + \nabla(f(E(X)))^T \cdot 0 \\ &= f(E(X)) \end{aligned}$$

$$E\Big[f(X)\Big] \geq f(E(X))$$

4

4. Let $f : \mathbb{R}^n \mapsto \mathbb{R}$ that is differentiable. Prove that f is convex if and only if

$$\langle \nabla f(\mathbf{x}) - \nabla f(\mathbf{y}), \mathbf{x} - \mathbf{y} \rangle \geq 0, \qquad \forall \mathbf{x}, \mathbf{y} \in \mathbb{R}^n.$$

Answer:

5

5. The Earth Mover's distance or Wasserstein metric is a distance or metric used to compare two probability distributions. See Levina and Bickel 2001 and references therein. Suppose we have two discrete probability distributions $\mathbf{p}$ and $\mathbf{q}$ on the integers $\{1, 2, \ldots, n\}$. So, p_i and q_i are both nonnegative for all $i \in \{1, \ldots, n\}$ and $\mathbf{p}^\mathsf{T}\mathbf{1} = \mathbf{q}^\mathsf{T}\mathbf{1} = 1$, where $\mathbf{1}$ is a vector all ones of length n. Then we can quantify the distance between $\mathbf{p}$ and $\mathbf{q}$ by the least amount of work it takes to reshape the distribution $\mathbf{p}$ into the distribution $\mathbf{q}$. Let $d(\mathbf{p}, \mathbf{q})$ denote the Earth Mover's distance between $\mathbf{p}$ and $\mathbf{q}$.

$$d(\mathbf{p}, \mathbf{q}) = \min_{f_{ij}} \sum_{i=1}^{n} \sum_{j=1}^{n} f_{ij} d_{ij},$$

subject to

$$\begin{aligned} f_{ij} &\geq 0, \quad \forall i, j \\ \sum_{j=1}^{n} f_{ij} &\leq p_i, \quad \forall i \\ \sum_{i=1}^{n} f_{ij} &\leq q_j, \quad \forall j \\ \sum_{i=1}^{n}\sum_{j=1}^{n} f_{ij} &= 1. \end{aligned}$$

The d_{ij} are given. These are non-negative distances between i and j. The f_{ij} quantify how much probability we are moving from p_i into q_j. Thus, the product $f_{ij}d_{ij}$ is the amount of work it takes to move probability mass from p_i into q_j. Show that the Earth Mover's distance $d(\mathbf{p}, \mathbf{q})$ is convex in $(\mathbf{p}, \mathbf{q}) \in [0, 1]^{2n}$.

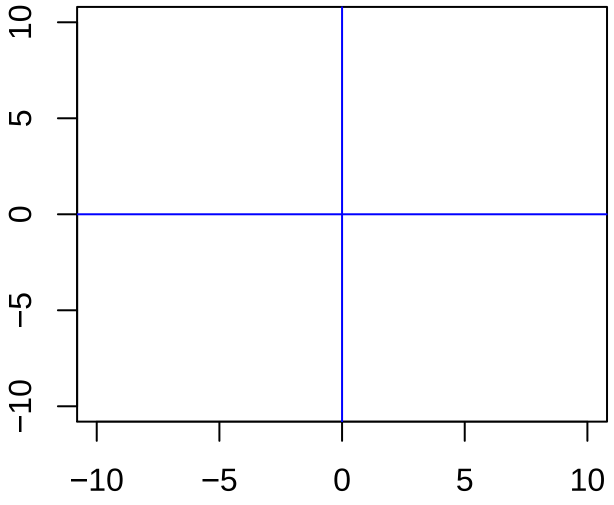
Answer:

6

6. What is the convex kernel of the set $A = \{(x, y) : x > 0, y = 0\} \cup \{(x, y) : x = 0, y > 0\}$. Be sure to justify your answer.

Answer:

Notice that A looks like the following.



The convex kernel of A is $D = \{x \in C : (x, y) \subset C, \forall y \in C\}$ where $(x, y) = \{z : z = \alpha x + (1-\alpha)y, \forall \alpha \in (0, 1)\}$. In words, d is in the convex kernel of A if a line can be drawn from d to and point in A and the whole line is in A. Thus, the only point in the convex kernel of A is $(0, 0)$.

Part 2

Part 2. Convexity-Checker

Recall that a function $f : R^n \mapsto R$ is convex if and only if $g(t) = f(x + tv)$ is convex for all $x + tv$ in dom f. In other words, a multivariate function is convex if and only if its univariate restrictions are convex.

The first function to go into your R package will leverage this fact to allow you to get some idea of the convexity of a multivariate function. The basic idea is that it will plot random univariate restrictions. Of course to prove convexity of a function all univariate restrictions need to be convex, and you can't plot all univariate restrictions. But if you plot several random restrictions and they all look bowl shaped, then it might be worth your time to prove the convexity of the function. Alternatively, if just one random restriction is not convex, then you know the multivariate function cannot be convex.

Please complete the following steps.

Step 1: Make an R package entitled "ST790_your_unity_id".

Step 2: Write a function "plot_restrictions."

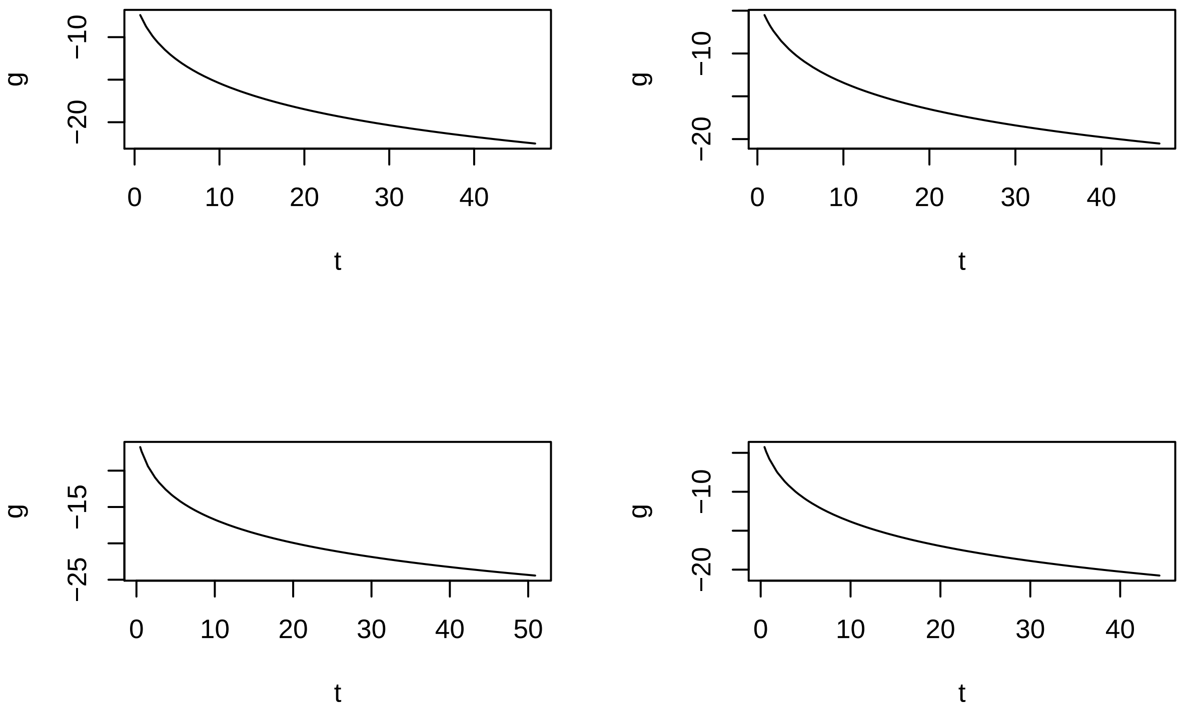
```r
#' @param fx handle to function in question
#' @param rx handle to function that returns a sequence of feasible sequence t and x + tv
#' @param nRow number of row plots
#' @param nCol number of column plots
plot_restrictions <- function(fx, rx, nRow=3, nCol=3) {

}
```

Step 3: Use your plot_restrictions function to verify the convexity of 9 random univariate restrictions of $f(X) = -\log \det X$ where X is a positive definite 5-by-5 matrix. Feel free to use the following functions for Step 3.

```r
fx <- function(X) {
  if (nrow(X) != ncol(X)) stop("'X' must be a square matrix")
  detX <- det(X)
  if (detX <= 0) stop("'X' must be positive definite")
  return(-log(det(X)))
}

rx <- function(n = 5, nt = 1e2) {
  ## Create positive definite X
  svdS <- svd(matrix(rnorm(n**2),n,n))
  U <- svdS$u
  X <- U %*% diag(1+rnorm(n)**2) %*% t(U)

  ## Create positive definite V
  svdS <- svd(matrix(rnorm(n**2),n,n))
  U <- svdS$u
  V <- U %*% diag(1+rnorm(n)**2) %*% t(U)

  ## Create sequence positive increasing sequence t and Z
  Z <- vector(mode="list", length=nt)
  t <- cumsum(runif(nt))
  for (i in 1:nt) {
    Z[[i]] <- X + t[i]*V
  }
  return(list(t=t, Z=Z))
}
```

Hint: You need to write something that does the following script for generic inputs.

```
nRow <- nCol <- 2
par(mfrow=c(nRow, nCol))
for (i in 1:(nRow*nCol)) {
  rand_rest <- rx()
  t <- rand_rest$t
  Z <- rand_rest$Z
  nt <- length(t)
  g <- double(nt)
  for (i in 1:nt) {
    g[i] <- fx(Z[[i]])
  }
  plot(t,g,type='l')
}
```


g
−10
−20
0 10 20 30 40
t
g
−10
−20
0 10 20 30 40
t
g
−15
−25
0 10 20 30 40 50
t
g
−10
−20
0 10 20 30 40
t