

Please always include this title page with your PDF. Include your name above.

- Submit your work in Gradescope as a PDF - you will identify where your "questions are."
- Identify the question number as you submit. Since we grade "blind" if the questions are NOT identified, the work WILL NOT BE GRADED and a 0 will be recorded. Always leave enough time to identify the questions when submitting.
- One section per page (if a page or less) - We prefer to grade the main solution in a single page, extra work can be included on the following page.
- Long instructions may be removed to fit on a single page.
- **Do not start a new question in the middle of a page.**
- Solutions to book questions are provided for reference.
- You may NOT submit given solutions - this includes minor modifications - as your own.
- Solutions that do not show individual engagement with the solutions will be marked as no credit and can be considered a violation of honor code.
- If you use the given solutions you must reference or explain how you used them, in particular...

For full credit, EACH book exercise in the Study Guides must use one or more of the following methods and FOR EACH QUESTION. Identify the number the method by number to ensure full credit.

Method 1 - Provide original examples which demonstrate the ideas of the exercise in addition to your solution.

Method 2 - Include and discuss the specific topics needed from the chapter and how they relate to the question.

Method 3 - Include original Python code, of reasonable length (as screenshot or text) to show how the topic or concept was explored.

Method 4 - Expand the given solution in a significant way, with additional steps and comments. All steps are justified. This is a good method for a proof for which you are only given a basic outline.

Method 5 - Attempt the exercise without looking at the solution and then the solution is used to check work. Words are used to describe the results.

Method 6 - Provide an analysis of the strategies used to understand the exercise, describing in detail what was challenging, who helped you or what resources were used. The process of understanding is described.

1. (20 pts) Reading the book carefully is essential in this class and in all advanced mathematics. This is an exercise in annotation.

For this first exercise, pick a section or page from Chapter one of VMLS.

Read straight through the section once for an overview. Include the following 3 items for #1.

- Write down your initial thoughts, questions, and first impressions ('whaaat???)
- Now, return to the section and slowly work through each line using a pencil or pen.
 - Expand equations.
 - Identify key concepts and explain in your own words.
 - Make note - is that a vector or a scalar?
 - Fill in missing ideas or steps.
 - Include a screenshot of your annotation.
- How has your understanding of the section changed?

Include a screenshot of your annotation.

1.4 Inner product

The (standard) *inner product* (also called *dot product*) of two n -vectors is defined as the scalar

$$a^T b = a_1 b_1 + a_2 b_2 + \dots + a_n b_n,$$

the sum of the products of corresponding entries. (The origin of the superscript 'T' in the inner product notation $a^T b$ will be explained in chapter 6.) Some other notations for the inner product (that we will not use in this book) are $\langle a, b \rangle$, $\langle a | b \rangle$, (a, b) , and $a \cdot b$. (In the notation used in this book, (a, b) denotes a stacked vector of length $2n$.) As you might guess, there is also a vector *outer product*, which we will encounter later, in §10.1. As a specific example of the inner product, we have

6 His $\begin{bmatrix} -1 \\ 2 \\ 2 \end{bmatrix}^T \begin{bmatrix} 1 \\ 0 \\ -3 \end{bmatrix} = (-1)(1) + (2)(0) + (2)(-3) = -7.$ 1 #

When $n = 1$, the inner product reduces to the usual product of two numbers.

Properties. The inner product satisfies some simple properties that are easily verified from the definition. If a , b , and c are vectors of the same size, and γ is a scalar, we have the following.

- **Commutativity.** $a^T b = b^T a$. The order of the two vector arguments in the inner product does not matter.
- **Associativity with scalar multiplication.** $(\gamma a)^T b = \gamma(a^T b)$, so we can write both as $\gamma a^T b$.
- **Distributivity with vector addition.** $(a+b)^T c = a^T c + b^T c$. The inner product can be distributed across vector addition.

These can be combined to obtain other identities, such as $a^T(\gamma b) = \gamma(a^T b)$, or $a^T(b + \gamma c) = a^T b + \gamma a^T c$. As another useful example, we have, for any vectors a, b, c, d of the same size,

$$(a + b)^T(c + d) = a^T c + a^T d + b^T c + b^T d.$$

Something that was interesting that I noticed when reading the textbook about the fundamentals of vectors is that when you have 2 vectors (v_1 , and v_2) transposed, the dot product is a singular number, which means that information is definitely lost over time.

So this led me to think that (for example) if you keep transposing the v_1 with the result of (v_1 transposed v_2), then you'll eventually get a super large number that isn't really reflective of what the original information entails

My gut reaction when seeing vectors for the first time was that they looked like sets and so when I see two vectors transposed next to each other, my brain jumps to "we should find the similarities of the two to form a new set" instead of seeing that the result of a transpose is 1 singular number.

I was convinced to believe this is the case when I read this "properties" section to realize that this transpose operation behaves similarly to how multiplication works in arithmetic. But what was even trickier was that the order of operation doesn't really matter in multiplication, but does matter for transpose because being mathematically the same does not mean the operation is computationally the same. Thomas does a good job of explaining this in the lecture when he talks about the # of operations changing due to the ordering of choosing when to transpose.

It took a little getting used to, but reframing the way I thought about this helped my understanding of section 1.4

2. (10pts) Solve the Random exercise from the video and Piazza in your own words here.

1.9 Symptoms vector. A 20-vector s records whether each of 20 different symptoms is present in a medical patient, with $s_i = 1$ meaning the patient has the symptom and $s_i = 0$ meaning she does not. Express the following using vector notation.

- (a) The total number of symptoms the patient has.
- (b) The patient exhibits five out of the first ten symptoms.

a) The total number of symptoms is a simple for-loop python with a counter looking something that represents something like: for every element in the array/vector, increment +1 to the counter so the counter ends up being a sum of all the elements in the vector so mathematically, it can be represented as the sum of # of elements inside the 20-vector record "s". That's where Thomas (the lecturer in the video) gets the Sigma notation summing from $i = 1$ to 20 incrementing based on if $s_{\text{sub-}i}$ exists. $s^T \mathbf{1}$ is just as correct as $s^T \mathbf{1}$ because transpose has commutative property.

```
In [5]: record_of_symptoms = [1, 0, 1, 0, 1, 1, 1, 0, 0, 0, 1, 1, 0, 1, 1, 0, 1, 0, 1, 1] # random example of positive/negative symptoms
number_of_symptoms = 0
for i in record_of_symptoms:
    number_of_symptoms += 1
print(number_of_symptoms)
20
a.k.a len(record_of_symptoms)
```

$$s^T \mathbf{1} = \sum_{i=1}^{20} s_i$$

b) If we want to represent that the sum of the first 10 elements $s = 5$, then 20 in the first sigma notation turns into "10", and $i = 1$ (still incrementing by 1 still) and $s_{\text{sub-}i} = 5$

in summation notation: $\sum_{i=1}^{10} s_i = 5$

We change the length of the record of symptoms from 20 -> 10 because we no longer care about iterating through the entire list of vectors, and only want to evaluate the first 10. We change $record_of_symptoms[i] = 5$ because problem b states that the positive symptoms must be 5 (as a given). So now we're given the end result and now have to write the transpose notation for it, which would be some vector "m" transpose $s = 5$. $m_{\text{sub-}i}$ is representative of a 20-vector defined as 1 if and only if i is greater than 1, and less than 10 (we're referring "i" as index in the list) and 0 otherwise (which means $m_{\text{sub-}i}$ is "0" when we are at the 11th index).

So to describe this is:

$$m^T \mathbf{5} \quad \text{or} \quad \sum_{i=1}^{10} s_i = 5$$

3. (10 pts) Explain the solution to 1.8 here in your own words. (Since you are given a solution, you will be graded on your ability to explain).

1.8 Profit and sales vectors. A company sells n different products or items. The n -vector p gives the profit, in dollars per unit, for each of the n items. (The entries of p are typically positive, but a few items might have negative entries. These items are called *loss leaders*, and are used to increase customer engagement in the hope that the customer will make other, profitable purchases.) The n -vector s gives the total sales of each of the items, over some period (such as a month), i.e., s_i is the total number of units of item i sold. (These are also typically nonnegative, but negative entries can be used to reflect items that were purchased in a previous time period and returned in this one.) Express the total profit in terms of p and s using vector notation.

The problem is asking for total profit, which is the multiplication of # of items sold, and profit per item. p sub- i is the profit of the i -th item, and s sub- i is the total sale of the i -th item, which means $(p$ sub- i)*(s sub- i) gives you the total profit for the i -th item (which is just 1 item in the sequence of items). So the total profit is to add the profit of each of these items together until you reach n -items (n = all the items).

The verbiage in this problem is tricky because you think of being done when you multiply p^* 's because multiplying profit and how many items sold gives you the total profit, right? In this case, no because that's just 1 item in the potential n -catalog of items. So let's say you sell utensils. p^* 's might only represent your profit selling 1 utensil (forks), but does not include the number of knives and spoons you've sold. The summation notation adapts it for all profit for all different kinds of items so we're adding the p^* 's of forks with the p^* 's of spoons and the p^* 's of knives (and etc.) until it captures all the items you've profited.

$$\text{Total profit} = \sum_{i=1}^n p_i s_i = p^T s$$

p transpose s where this is the dot product to represent the profit vector of p relative to sales vector of s

4. (10 pts) Explain the solution to 1.16 here in your own words. (Since you are given a solution, you will be graded on your ability to explain).

1.16 *Inner product of nonnegative vectors.* A vector is called *nonnegative* if all its entries are nonnegative.

- (a) Explain why the inner product of two nonnegative vectors is nonnegative.

Let a and b be two n -dimensional non-negative vectors, meaning that $a_{\text{sub-}i} \geq 0$ and $b_{\text{sub-}i} \geq 0$ for $i = 1, 2, \dots$ until n . The inner product of a and b is a transpose b . Each term at $a_{\text{sub-}i}$ and $b_{\text{sub-}i}$ in the summation notation is going to be the product of two non-negative numbers, which results in a non-negative number. Then when you add the non-negative terms together, since all the inputs are non-negative, this will also result in a non-negative number so that the inner product is going to be non-negative.

- (b) Suppose the inner product of two nonnegative vectors is zero. What can you say about them? Your answer should be in terms of their respective sparsity patterns, i.e., which entries are zero and nonzero.

If a transpose $b = 0$, then for each i $(a_{\text{sub-}i})^*(b_{\text{sub-}i})$ is ≥ 0 . The only way their sum can be zero is if $(a_{\text{sub-}i})^*(b_{\text{sub-}i}) = 0$ for every i . This means that for each i , at least one of $a_{\text{sub-}i}$ or $b_{\text{sub-}i}$ must be zero. In terms of sparsity, the nonzero entries of a and b must occur disjointly, in other words, if $a_{\text{sub-}i} > 0$, then $b_{\text{sub-}i}$ has to be $= 0$ and vice versa.

If the inner product is zero, the nonzero entries of the 2 vectors cannot overlap with each other which means that their nonzero elements must be disjointed.

To give an example with tug of war where there's team A and team B. Team A = [1, 0, 0, 1, 0] where 1st and 4th person is pulling with all their strength and team B = [0, 1, 1, 0, 1] where 2nd, 3rd, 5th person is pulling with all their strength. When team A pulls, but team B does nothing, their contribution is $1*0 = 0$. When Team B pulls but team A doesn't do anything, their contribution is $0*1 = 0$. So the inner product ends up being a transpose $b = 0$. The two teams never get to oppose each other at the same spot and the pulling efforts never overlap, so when one team pulls, the other is doing nothing.

5. (50 points) Create a Jupyter Notebook of the following:

The Python **Companion** (linked at the top of the course) includes examples of code that can help you throughout the course and provides the basis for the fundamentals of vector operations we will use throughout the course.

Every week you can use these code elements to illustrate concepts and ideas from the weeks.

This week we guide you through an example.

1. Find the Python Companion and [sections 1.1 - 1.4](#),
2. Select at least 10 code blocks (you may want to do all of them) that look useful.
3. Rewrite (best) or copy and paste each code cell **into your own Jupyter Notebook**.
4. Add notes in your own words in a text block above about the code using the notes given AND incorporate ideas from the book (cite the book page).
5. For each section of code include at least one additional specific example of using the code.
6. **Attach a PDF of your Jupyter Notebook here** to include in the single PDF study guide.