3.250. Color perception. Human color perception is based on the responses of three different types of color light receptors, called cones. The three types of cones have different spectral-response characteristics, and are called L, M, and, S because they respond mainly to long, medium, and short wavelengths, respectively. In this problem we will divide the visible spectrum into 20 bands, and model the cones’ responses as follows:

\[
L_{\text{cone}} = \sum_{i=1}^{20} l_i p_i, \quad M_{\text{cone}} = \sum_{i=1}^{20} m_i p_i, \quad S_{\text{cone}} = \sum_{i=1}^{20} s_i p_i,
\]

where \( p_i \) is the incident power in the \( i \)th wavelength band, and \( l_i, m_i \) and \( s_i \) are nonnegative constants that describe the spectral responses of the different cones. The perceived color is a complex function of the three cone responses, i.e., the vector \((L_{\text{cone}}, M_{\text{cone}}, S_{\text{cone}})\), with different cone response vectors perceived as different colors. (Actual color perception is a bit more complicated than this, but the basic idea is right.)

a) Metamers. When are two light spectra, \( p \) and \( \tilde{p} \), visually indistinguishable? (Visually identical lights with different spectral power compositions are called metamers.)

b) Visual color matching. In a color matching problem, an observer is shown a test light, and is asked to change the intensities of three primary lights until the sum of the primary lights looks like the test light. In other words, the observer is asked the find a spectrum of the form

\[ p_{\text{match}} = a_1 u + a_2 v + a_3 w, \]

where \( u, v, w \) are the spectra of the primary lights, and \( a_i \) are the intensities to be found, that is visually indistinguishable from a given test light spectrum \( p_{\text{test}} \). Can this always be done? Discuss briefly.

c) Visual matching with phosphors. A computer monitor has three phosphors, \( R, G, \) and \( B \). It is desired to adjust the phosphor intensities to create a color that looks like a reference test light. Find weights that achieve the match or explain why no such weights exist. The data for this problem is in color_perception_data.json, which contains the vectors wavelength, B_phosphor, G_phosphor, R_phosphor, L_coefficients, M_coefficients, S_coefficients, and test_light.

d) Effects of illumination. An object’s surface can be characterized by its reflectance (i.e., the fraction of light it reflects) for each band of wavelengths. If the object is illuminated with a light spectrum characterized by \( I_i \), and the reflectance of the object is \( r_i \) (which is between 0 and 1), then the reflected light spectrum is given by \( I_i r_i \), where \( i = 1, \ldots, 20 \) denotes the wavelength band. Now consider two objects illuminated (at different times) by two different light sources, say an incandescent bulb and sunlight. Sally argues that if the two objects look identical when illuminated by a tungsten bulb, then they will look identical when illuminated by sunlight. Beth disagrees: she says that two objects can appear identical when illuminated by a tungsten bulb, but look different when lit by sunlight. Who is right? If Sally is right, explain why. If Beth is right give an example
of two objects that appear identical under one light source and different under another. You can use the vectors \textit{sunlight} and \textit{tungsten} defined in the data file as the light sources.

\textbf{Remark.} Spectra, intensities, and reflectances are all nonnegative quantities, which the material of EE263 doesn’t address. So just ignore this while doing this problem. These issues can be handled using the material of EE364a, however.

3.340. \textbf{Vector spaces over the Boolean field.} In this course the \textit{scalar field}, \textit{i.e.}, the components of vectors, will usually be the real numbers, and sometimes the complex numbers. It is also possible to consider vector spaces over other fields, for example \( \mathbb{Z}_2 \), which consists of the two numbers 0 and 1, with Boolean addition and multiplication (\textit{i.e.}, \( 1 + 1 = 0 \)). Unlike \( \mathbb{R} \) or \( \mathbb{C} \), the field \( \mathbb{Z}_2 \) is finite, indeed, has only two elements. A vector in \( \mathbb{Z}_2^n \) is called a \textit{Boolean vector}. Much of the linear algebra for \( \mathbb{R}^n \) and \( \mathbb{C}^n \) carries over to \( \mathbb{Z}_2^n \). For example, we define a function \( f : \mathbb{Z}_2^n \rightarrow \mathbb{Z}_2^m \) to be linear (over \( \mathbb{Z}_2 \)) if

\[ f(x + y) = f(x) + f(y) \quad \text{and} \quad f(\alpha x) = \alpha f(x) \]

for every \( x, y \in \mathbb{Z}_2^n \) and \( \alpha \in \mathbb{Z}_2 \). It is easy to show that every linear function can be expressed as matrix multiplication, \textit{i.e.}, \( f(x) = Ax \), where \( A \in \mathbb{Z}_2^{m \times n} \) is a Boolean matrix, and all the operations in the matrix multiplication are Boolean, \textit{i.e.}, in \( \mathbb{Z}_2 \). Concepts like nullspace, range, independence and rank are all defined in the obvious way for vector spaces over \( \mathbb{Z}_2 \). Although we won’t consider them in this course, there are many important applications of vector spaces and linear dynamical systems over \( \mathbb{Z}_2 \). In this problem you will explore one simple example: block codes. \textit{Linear block codes.} Suppose \( x \in \mathbb{Z}_2^n \) is a Boolean vector we wish to transmit over an unreliable channel. In a linear block code, the vector \( y = Gx \) is formed, where \( G \in \mathbb{Z}_2^{m \times n} \) is the \textit{coding matrix}, and \( m > n \). Note that the vector \( y \) is ‘redundant’; roughly speaking we have \textit{coded} an \( n \)-bit vector as a (larger) \( m \)-bit vector. This is called an \( (n, m) \) code. The coded vector \( y \) is transmitted over the channel; the received signal \( \hat{y} \) is given by

\[ \hat{y} = y + v, \]

where \( v \) is a noise vector (which usually is zero). This means that when \( v_i = 0 \), the \( i \)th bit is transmitted correctly; when \( v_i = 1 \), the \( i \)th bit is changed during transmission. In a \textit{linear decoder}, the received signal is multiplied by another matrix: \( \hat{x} = H\hat{y} \), where \( H \in \mathbb{Z}_2^{n \times m} \). One reasonable requirement is that if the transmission is perfect, \textit{i.e.}, \( v = 0 \), then the decoding is perfect, \textit{i.e.}, \( \hat{x} = x \). This holds if and only if \( H \) is a left inverse of \( G \), \textit{i.e.}, \( HG = I_n \), which we assume to be the case.

a) What is the practical significance of range(\( G \))?

b) What is the practical significance of null(\( H \))?

c) A one-bit error correcting code has the property that for any noise \( v \) with one component equal to one, we still have \( \hat{x} = x \). Consider \( n = 3 \). Either design a one-bit error correcting linear block code with the smallest possible \( m \), or explain why it cannot be done. (By design we mean, give \( G \) and \( H \) explicitly and verify that they have the required properties.)

\textbf{Remark:} linear decoders are never used in practice; there are far better nonlinear ones.
3.540. **Sparse solution of underdetermined equations.** Suppose that \( y = Ax \), where \( A \in \mathbb{R}^{m \times n} \), with \( m < n \) (so these equations are underdetermined). You are given \( A \) and \( y \), but not \( x \). Without any further assumptions, you cannot determine \( x \). But now we add the additional information that \( x \) has \( k < n \) nonzero entries. You are told \( k \), the number of nonzero entries in \( x \), but not the particular indices of the entries of \( x \) that are nonzero. In some cases, it is possible to determine \( x \) (given the additional information that it has \( k \) nonzeros), even though the linear equations are underdetermined. (This is a basic problem in a fascinating area of current research called compressed sensing, compressive sampling, and several other names. Of course, you don’t need to know any of this research to solve this problem.)

Now consider the specific case with \( A, y, \) and \( k \) given in the file `underdet_sparse_data.json`. Choose one of the following.

a) **You can’t find \( x \).** To show this, find \( x \) and \( \tilde{x} \), not the same, each with \( k \) nonzero entries, which satisfy \( y = Ax = A\tilde{x} \).

b) **You can find \( x \).** Find \( x \), and verify that it satisfies \( y = Ax \), and has \( k \) nonzero entries. Explain how you know that there is no other \( \tilde{x} \), with \( k \) nonzero entries, that satisfies \( y = Ax \).

In either case, give the code that you use to verify that the required property holds (and in the second case, that the \( x \) you found is the only one).

Your solution to either problem can use any of the concepts and methods we have covered in the class so far: QR factorization, rank, range, nullspace, least-squares approximate solutions, and so on. Your solution can involve a loop or loops over a finite (and possibly large) number of calculations involving the ideas above.

4.580. **Projection matrices.** A matrix \( P \in \mathbb{R}^{n \times n} \) is called a projection matrix if \( P = P^T \) and \( P^2 = P \).

a) Show that if \( P \) is a projection matrix then so is \( I - P \).

b) Suppose that the columns of \( U \in \mathbb{R}^{n \times k} \) are orthonormal. Show that \( UU^T \) is a projection matrix. (Later we will show that the converse is true: every projection matrix can be expressed as \( UU^T \) for some \( U \) with orthonormal columns.)

c) Suppose \( A \in \mathbb{R}^{n \times k} \) is full rank, with \( k \leq n \). Show that \( A(A^TA)^{-1}A^T \) is a projection matrix.

d) Show that \( x - Px \) is orthogonal to \( \text{range}(P) \) for any \( x \). (Aside: this means that that \( Px \) is the closest point in \( \text{range}(P) \) to \( x \).)

4.670. **Solving linear equations via QR factorization.** Consider the problem of solving the linear equations \( Ax = y \), with \( A \in \mathbb{R}^{n \times n} \) nonsingular, and \( y \) given. We can use the Gram-Schmidt procedure to compute the QR factorization of \( A \), and then express \( x \) as \( x = A^{-1}y = R^{-1}(Q^Ty) = R^{-1}z \), where \( z = Q^Ty \). In this exercise, you’ll develop a method for computing \( x = R^{-1}z \), i.e., solving \( Rx = z \), when \( R \) is upper triangular and nonsingular (which means its diagonal entries are all nonzero).
The trick is to first find $x_n$; then find $x_{n-1}$ (remembering that now you know $x_n$); then find $x_{n-2}$ (remembering that now you know $x_n$ and $x_{n-1}$); and so on. The algorithm you will discover is called back substitution, because you are substituting known or computed values of $x_i$ into the equations to compute the next $x_i$ (in reverse order). Be sure to explain why the algorithm you describe cannot fail.