REVIEW SHEET FOR MIDTERM 1: BASIC

MATH 196, SECTION 57 (VIPUL NAIK)

We will not be going over this sheet, but rather, we'll be going over the advanced review sheet in the session. Please review this sheet on your own time.

The summaries here are identical with the executive summaries you will find at the beginning of the respective lecture notes PDF files. The summaries are not intended to be exhaustive. Please review the original lecture notes as well, especially if any point in the summary is unclear. The section titles have the same names as the names of the corresponding lecture notes.

1. Linear functions: a primer

Words ...

(1) Linear models arise both because some natural and social phenomena are intrinsically linear, and because they are computationally tractable and hence desirable as approximations, either before or after logarithmic and related transformations.
(2) Linear functions (in the affine linear sense) can be characterized as functions for which all the second-order partial derivatives are zero. The second-order pure partial derivatives being zero signifies linearity in each variable holding the others constant (if this condition is true for each variable separately, we say the function is (affine) multilinear). The mixed partial derivatives being zero signifies *additive separability* in the relevant variables. If this is true for every pair of input variables, the function is completely additively separable in the variables.
(3) We can use logarithmic transformations to study multiplicatively separable functions using additively separable functions. For a few specific functional forms, we can make them linear as well.
(4) We can use the linear paradigm in the study of additively separable functions where the components are known in advance up to scalar multiples.
(5) If a function type is linear in the parameters (not necessarily in the input variables) we can use (input,output) pairs to obtain a system of linear equations in the parameters and determine the values of the parameters. Note that a linear function of the variables with no restrictions on the coefficients and intercepts must also be linear in the parameters (with the number of parameters being one more than the number of variables). However, there are many nonlinear functional forms, such as polynomials, that are linear in the parameters but not in the variables.
(6) Continuing the preceding point, the number of well-chosen (input,output) pairs that we need should be equal to the number of parameters. Here, the "well-chosen" signifies the absence of dependencies between the chosen inputs. However, choosing the bare minimum number does not provide any independent confirmation of our model. To obtain independent confirmation, we should collect additional (input,output) pairs. The possibility of modeling and measurement errors may require us to introduce error-tolerance into our model, but that is beyond the scope of the current discussion. We will return to it later.

Actions ...

(1) One major stumbling block for people is in writing the general functional form for a model that correctly includes parameters to describe the various degrees of freedom. Writing the correct functional form is half the battle. It's important to have a theoretically well-grounded choice of functional form and to make sure that the functional form as described algebraically correctly describes what we have in mind.
(2) It's particularly important to make sure to include a parameter for the *intercept* (or *constant term*) unless theoretical considerations require this to be zero.

(3) When dealing with polynomials in multiple variables, it is important to make sure that we have accounted for all possible monomials.

(4) When dealing with piecewise functional descriptions, we have separate functions for each piece interval. We have to determine the generic functional form for each piece. The total number of parameters is the sum of the number of parameters used for each of the functional forms. In particular, if the nature of the functional form is the same for each piece, the total number of parameters is (number of parameters for the functional form for each piece) $\times$ (number of pieces).

2. Equation-solving with a special focus on the linear case: a primer

Words ...

(1) We need to solve equations both for determining the parameters in a general functional description of a model, and for using existing models to estimate the values of real-world variables. Equations are also useful in solving max/min problems.

(2) When evaluating equation-solving algorithms for suitability, we should consider time, memory, parallelizability, precision, and elegance.

(3) The dimension of the solution space to a system of equations that has no redundancies or inconsistencies is expected to be (number of variables) - (number of equations). If there are fewer equations than variables, we expect that the system is *underdetermined.* If there are more equations than variables, we expect that the system is *overdetermined.* Generally, this means that either some of the equations are redundant (i.e., they do not provide additional information) or the equations are inconsistent.

(4) For diagonal systems of equations, i.e., systems of equations where each equation involves only one variable, we can solve the equations separately, then string together the coordinates of the solution. The process is parallelizable. The solution set satisfies a *rectangular completion property* (note: this isn't a standard term, it's just a shorthand for something that's hard to describe without full symbolism; you might prefer to think of it as a *coordinate interchange property* if that name connects better with the concept for you).

(5) For triangular systems of equations, we need to solve the equations in a particular order, and if we do so, we have to solve a series of equations in one variable. The solution set need not satisfy the rectangular completion property. In case of multiple solutions, we need to make cases and branch each case. Parallelization is possible between the branches but not between the equations themselves.

(6) The same remarks apply for triangular systems of linear equations, except that there is no branching to worry about.

(7) When we manipulate equations (by adding, subtracting, and multiplying) to obtain new equations, we must remember to keep the old equations around to protect against loss of information. However, keeping *all* the old equations around can result in a memory overload. The general idea is to keep enough of the old equations around that the other old equations that we are discarding can be recovered from the new system. In other words, we want our transformations to be *reversible.*

(8) One manipulation technique is to use one of the equations to eliminate one of the variables by expressing it in terms of the other variables. If we are able to do this systematically all the way through, we would have reduced the original system to a triangular system.

(9) Another manipulation technique is to add and subtract multiples of one equation to another equation. We generally keep the equation that is being multiplied before the addition, and discard the equation that is not being multiplied by anything, because that guarantees reversibility. If the value we are multiplying with is a nonzero constant, we can also keep the other equation instead.

(10) We typically add and subtract equations with the explicit goal of eliminating variables or eliminating difficult expressions in terms of the variables.

Actions ...

(1) Please be sure not to throw out information when manipulating the system. When in doubt, store more rather than less.

(2) Please remember that, as soon as you discover an inconsistency, you can abandon the equation-solving attempt (or the branch you are in). All previous solutions that you have found don't amount to

anything. In a diagonal system, if any one equation in the system has no solution for its corresponding variable, the system as a whole has no solution. In a triangular system, as soon as we discover an equation that has no solution, discard the branch (from the most recent branching point onward).

3. Gauss-Jordan elimination: a method to solve linear systems

Words and actions combined ...

(1) A system of linear equations can be stored using an *augmented matrix*. The part of the matrix that does not involve the constant terms is termed the *coefficient matrix*.
(2) Variables correspond to columns and equations correspond to rows in the coefficient matrix. The augmented matrix has an extra column corresponding to the constant terms.
(3) In the paradigm where the system of linear equations arises from an attempte to determine the parameters of a model (that is linear in the parameters) using (input,output) pairs, the parameters of the model are the new variables, and therefore correspond to the columns. The (input,output) pairs correspond to the equations, and therefore to the rows. The input part controls the part of the row that is in the coeficient matrix, and the output part controls the augmenting entry.
(4) If the coefficient matrix of a system of simultaneous linear equations is in reduced row-echelon form, it is easy to read the solutions from the coefficient matrix. Specifically, the non-leading variables are the parameters, and the leading variables are expressible in terms of those.
(5) In reduced row-echelon form, the system is inconsistent if and only if there is a row of the coefficient matrix that is all zeros with the corresponding augmented entry nonzero. Note that if the system is *not* in reduced row-echelon form, it is *still* true that a zero row of the coefficient matrix with a nonzero augmenting entry implies that the system is inconsistent, but the converse does not hold: the system may well be inconsistent despite the absence of such rows.
(6) In reduced row-echelon form, if the system is consistent, the dimension of the solution space is the number of non-leading variables, which equals (number of variables) - (number of nontrivial equations). Note that the all zero rows give no information.
(7) Through an appropriately chosen sequence of row operations (all reversible), we can transform any linear system into a linear system where the coefficient matrix is in reduced row-echelon form. The process is called Gauss-Jordan elimination.
(8) The process of Gauss-Jordan elimination happens entirely based on the coefficient matrix. The final column of the augmented matrix is affected by the operations, but does not control any of the operations.
(9) There is a quicker version of Gauss-Jordan elimination called Gaussian elimination that converts to row-echelon form (which is triangular) but not reduced row-echelon form. It is quicker to arrive at, but there is more work getting from there to the actual solution. Gaussian elimination is, however, sufficient for determining which of the variables are leading variables and which are non-leading variables, and therefore for computing the dimension of the solution space and other related quantities.
(10) The arithmetic complexity of Gauss-Jordan elimination, in both space and time terms, is polynomial in n. To get a nicely bounded bit complexity (i.e., taking into account the sizes of the numbers blowing up) we need to modify the algorithm somewhat, and we then get a complexity that is jointly poynomial in n and the maximum bit length.
(11) Depending on the tradeoff between arithmetic operations and using multiple processors, it is possible to reduce the arithmetic complexity of Gauss-Jordan elimination considerably.
(12) Gauss-Jordan elimination is not conceptually different from iterative substitution. Its main utility lies in the fact that it is easier to code since it involves only numerical operations and does not require the machine to understand equation manipulation. On the other hand, because the operations are purely mechanical, it may be easier to make careless errors because of the lack of "sense" regarding the operations. To get the best of both worlds, use the Gauss-Jordan elimination procedure, but keep the symbolic algebra at the back of your mind while doing the manipulations.
(13) The Gauss-Jordan elimination process is only one of many ways to get to reduced row-echelon form. For particular structures of coefficient matrices, it may be beneficial to tweak the algorithm a little

(for instance, swapping in an easier row to the top) and save on computational steps. That said, the existence of the Gauss-Jordan elimination process gives us a guarantee that we can reach our goal by providing one clear path to it. If we can find a better path, that's great.

(14) We can use the Euclidean algorithm/gcd algorithm/Bareiss algorithm. This is a variant of the Gauss-Jordan algorithm that aims for the same end result (a reduced row-echelon form) but chooses a different sequencing and choice of row operations. The idea is to keep subtracting multiples of rows from one another to get the entries down to small values (hopefully to 1), rather than having to divide by the leading entry. It's not necessary to master this algorithm, but you might find some of its ideas helpful for simplifying your calculations when solving linear systems.

4. Linear systems and matrix algebra

Words and actions combined ...

(1) The *rank* of a matrix is defined as the number of nonzero rows in its reduced row-echelon form, and is also equal to the number of leading variables. The rank of a matrix is less than or equal to the number of rows. It is also less than or equal to the number of columns.
(2) The rank of the coefficient matrix of a system of simultaneous linear equations describes the number of independent equational constraints in the system.
(3) How far the coefficient matrix is from having full column rank determines the dimension of the solution space if it exists.
(4) How far the coefficient matrix is from having full row rank determines the probability that the system is consistent, roughly speaking. If the coefficient matrix has full row rank, then the system is consistent for all outputs. Otherwise, it is consistent only for some outputs and inconsistent for others.
(5) For a consistent system, the dimension of the solution space equals (number of variables) - (rank).
(6) There is a concept of "expected dimension" which is (number of variables) - (number of equations). Note that if the system does not have full row rank, the expected dimension is less than the actual dimension (if consistent). The expected dimension can be thought of as averaging the actual dimensions over all cases, where inconsistent cases are assigned dimensions of $-\infty$. This is hard to formally develop, so we will leave this out.
(7) There are various terms commonly associated with matrices: zero matrix, square matrix, diagonal, diagonal matrix, scalar matrix, identity matrix, upper-triangular matrix, and lower-triangular matrix.
(8) A vector can be represented as a row vector or a column vector.
(9) We can define a dot product of two vectors and think of it in terms of a sliding snake.
(10) We can define a matrix-vector product: a product of a matrix with a column vector. The product is a column vector whose entries are dot products of the respective rows of the matrix (considered as vectors) with the column vector.
(11) Matrix-vector multiplication is linear in the vector.
(12) A linear system of equations can be expressed as saying that the coefficient matrix times the input vector column (this is the column of unknowns) equals the output vector column (this is the column that would be the last column in the augmented matrix).

5. Hypothesis testing, rank, and overdetermination

Words ...

(1) In order to test a hypothesis, we must conduct an experiment that could conceivably come up with an outcome that would falsify the hypothesis. This relates to Popper's notion of falsifiability.
(2) In the setting where we use a model with a functional form that is linear in the parameters, the situation where the coefficient matrix (dependent on the inputs) does *not* have full row rank is the situation where we can use consistency of the system to obtain additional confirmation of the hypothesis that the model is correct. If the coefficient matrix has full column rank, we can determine the parameters uniquely assuming consistency. The ideal situation would be that we choose inputs

such that the coefficient matrix has full column rank but does not have full row rank. In this situation, we can obtain verification of the hypothesis *and* find the parameter values.

(3) In order to test a hypothesis of a function of multiple variables being affine linear, we could choose three points that are collinear in the input space and see if the outputs behave as predicted. If they do, then this is evidence in favor of linearity, but it is not conclusive evidence. If they do not, this is conclusive evidence against linearity.

(4) If the goal is to find the coefficients rather than to test the hypothesis of linearity, we should try picking independent inputs (in general, as many inputs as the number of parameters, which, for an affine linear functional form, is one more than the number of variables). Thus, the choice of inputs differ for the two types of goals. If, however, we are allowed enough inputs, then we can both find all the coefficients *and* test for linearity.

6. Linear transformations

Words and actions combined...

(1) A $n \times m$ matrix defines a function from $\mathbb{R}^m$ to $\mathbb{R}^n$ via matrix-vector multiplication. A function arising in this manner is termed a *linear transformation*. Every linear transformation arises from a unique matrix, i.e., there is a bijection between the set of $n \times m$ matrices and the set of linear transformations from $\mathbb{R}^m$ to $\mathbb{R}^n$.

(2) A function (also called map) $f : A \to B$ of sets is termed *injective* if no two elements of A map to the same element of B. It is termed *surjective* if B equals the range of f. It is termed *bijective* if it is both injective and surjective. A bijective map has a unique inverse map.

(3) The standard basis vector $\vec{e}_i$ is the vector with a 1 in the i^{th} coordinate and 0s elsewhere. The image of $\vec{e}_i$ is precisely the i^{th} column of the matrix describing the linear tranformation.

(4) A linear transformation can alternatively be defined as a map that preserves addition and scalar multiplication. Explicitly, $T : \mathbb{R}^m \to \mathbb{R}^n$ is a linear transformation if and only if $T(\vec{x} + \vec{y}) = T(\vec{x}) + T(\vec{y})$ (additivity) and $T(a\vec{x}) = aT(\vec{x})$ (scalar multiples) for all $\vec{x}, \vec{y} \in \mathbb{R}^m$ and all $a \in \mathbb{R}$.

(5) A linear transformation $T : \mathbb{R}^m \to \mathbb{R}^n$ is *injective* if the matrix of T has full column rank, which in this case means rank m, because the dimensions of the matrix are $n \times m$. Note that this in particular implies that $m \leq n$. The condition $m \leq n$ is a *necessary but not sufficient condition* for the linear transformation to be injective.

(6) A linear transformation $T : \mathbb{R}^m \to \mathbb{R}^n$ is *surjective* if the matrix of T has full row rank, which in this case means rank n, because the dimensions of the matrix are $n \times m$. Note that this in particular implies that $n \leq m$. The condition $n \leq m$ is a *necessary but not sufficient condition* for the linear transformation to be surjective.

(7) A linear transformation $T : \mathbb{R}^m \to \mathbb{R}^n$ is *bijective* if the matrix of T has full row rank and full column rank. Thus forces $m = n$, and forces the (now square) matrix to have full rank. As mentioned before, this is equivalent to invertibility.

(8) The inverse of a diagonal matrix with all diagonal entries nonzero is the matrix obtained by inverting each diagonal entry individually.

(9) A permutation matrix is a matrix where each row has one 1 and all other entries 0, and each column has one 1 and all other entries 0. A permutation matrix acts by permuting the standard basis vectors among themselves. It can be inverted by permuting the standard basis vectors among themselves in the reverse direction. Hence, the inverse is also a permutation matrix. If the permutation matrix just flips two basis vectors, it is self-inverse.

(10) The inverse of a shear operation is the shear operation obtained by using the negative of the shear. In particular:

$$\begin{bmatrix} 1 & 1 \\ 0 & 1 \end{bmatrix}^{-1} = \begin{bmatrix} 1 & -1 \\ 0 & 1 \end{bmatrix}$$

The last two points from the summary of the lecture notes are not directly relevant to the midterm, but are included for completeness below.

(11) Inverting a matrix requires solving a system of simultaneous linear equations with that as coefficient matrix and with the augmenting column a *generic* output vector, then using the expression for the input in terms of the output to get the matrix of the transformation. It can alternatively be done by augmenting the matrix with the identity matrix, row-reducing the matrix to the identity matrix, and then looking at what the augmented side has become.

(12) We can use linear transformations for the purposes of coding, compression, and extracting relevant information. In many of these practical applications, we are working, not over the real numbers, but over the field of two elements, which is ideally suited for dealing with the world of bits (binary digits).