Lecture 12 Script

12.1 Linear Algebra

Slide 1: Linear Algebra - Title Page

1. Algebraic neural networks are rooted in the theory of algebraic signal processing. This is a recasting of signal processing in the language of abstract linear algebra. We therefore must begin with some recollections. In this video we go over the foundational concepts of linear algebra: fields, vectors spaces and algebras.

2. We will use these definitions to point out that linear information processing is tantamount to the application of operators that live in the algebra of endomorphisms of a vector space.

Slide 2: Fields

1. The first definition that we recall is that of a field F. If we are not interested in being formal, a Field can be defined as a set in which a sum and a multiplication operation are defined.

2. Thus, the introduction of a particular field is intended to define numbers and the operations we perform on them. This is not sophisticated. We are referring about things like the set of real numbers. Or the set of complex numbers.

3. On which there are two operations defined. The sum and the product. Which have the usual meaning with which you are familiar since your kindergarten days.

4. If we want to be formal about it, a field is defined as a set on which two binary operations are defined: Addition and Multiplication. These operations must satisfy certain properties:

5. They must be associative operations

6. They must also be commutative

7. And they both must have identity elements. A zero for the sum. And a 1 for the multiplication.

8. The sum is required to have an inverse (or opposite) for all elements

9. And the multiplication is required to have an inverse for all elements, except the identity element of the product.

10. The final property of the field is the distributive property of the multiplication operation over the addition operation. It may look supercilious to be formal about the definition of something so simple as additions and multiplications. But it is not. The idea is to see that any set with a pair of operations that satisfy these properties is analogous to a set of numbers. This may include sets that do not look like numbers. And as long as the results we derive depend on the properties we list, they would hold true for these other sets that don't look like numbers but share these fundamental properties with them.

Slide 3: Vector Spaces

1. The second foundational concept of linear algebra is that of a vector space.

2. A vector space M over a field F is a set whose elements can be added together and that can also be multiplied by elements of the field. Very importantly, the elements of the vector space don't have to be multipliable. We don't require the definition of a product between elements of the vector space. We just require that the definition of a product between elements of the vector space and the field.

3. A vector space is a set of arrows.

4. For our purposes, the vector space defines the type of signals we want to process. They can be vectors in R-n. Or square integrable functions in the interval 0-1. Sequences are another example.

5. The difference between a field and a vector space, is the incorporation of two new operations.

6. The addition of signals. And the multiplication of signals by scalars.

7. Formally, A vector space over the field F is a set with two operations. Vector addition and scalar multiplication. These operations must satisfy the following properties:

8. The vector addition must be associative
9. Commutative
10. It must have an identity element. The all zero vector if you wish.
11. And it must have an inverse (or opposite).
12. Asides from these properties of the addition operation, we require four properties of the scalar multiplication.
13. We require it to be compatible with the field multiplication. This is a sort of associative property except that the operations involved are different. One of them is a filed multiplication. The other is a scalar multiplication.
14. The scalar multiplication must have an identity element.
15. And it must be distributive with respect to the vector addition.
16. And with respect to the field addition. As in the case of the field definition all of this looks supercilious. But it is intended to be general. We abstract the minimal number of properties that allows us to derive theorems that are as widely applicable as possible. This generality allows us to claim, for instance, that there is little difference between the linear processing of vectors, functions, and sequences.

Slide 4: Associative Algebras

1. The final definition we introduce is the notion of an Associative Algebra. For this one we don't need an informal definition, because the formal definition is sufficiently simple.
2. An associative algebra A is a vector space.
3. In which we also define a bilinear map that we denote as a product with an asterisk.
4. The only condition we impose in this product is that the product be associative.
5. If the algebra also has an identity element we say the algebra is an algebra with unity.

6. And if the algebra is such that the order of the products is not important, we say the algebra is commutative. We will work with associative algebras with unity. And, for the most part, with algebras that are commutative.

7. What the algebra has added to a vector space is a fifth operation. On top of the two fields operations and the two vector space operations. This operation is intended to represent the linear transformation of a signal. To explain this we need to introduce the space of endomorphisms of a vector space.

Slide 5: Signals

1. Before we do that, let's recap the important aspects of a vector space as they pertain to the linear processing of signals.

2. Signals, are the entities that we want to process. And in order to be able to process them, we require them to be elements x of a vector space M. Why do we want to make this request?

3. Because, at the very least, we want to be able to add two signals.

4. And we want to be able to scale signals with the elements of a field. The idea of defining a vector space formally is to isolate the properties that give meaning to these two operations.

5. In keeping the definition general we allow for the processing of vectors with n components. This means the use of R-n as the vector space n. Which is what we do to process graph signals. Or discrete signals.

6. But we can also refer to the processing of functions defined in the interval 0-1 with finite energy. This is the vector space of graphon signals.

7. There are many more sets of signals that are vector spaces. The most common are sequences, functions in R. And sequences with two indexes. These are models of discrete time, continuous time, and images, respectively.

Slide 6: Endomorphisms

1. The important operation that is missing from a vector space is the notion of a linear transformation. This is where algebras in general and endomorphisms in particular come into the picture.

2. An endomorphism e is simply a linear map from the vector space M into itself.

3. That the map e is linear means what you expect it to mean. Namely, that applying e to a linear combination of elements x_1 and x_2 of the vector space.

4. Results in the respective linear combination of the results of applying e to x_1 and x_2.

5. If we consider the vector space M equals R-n. The transformation e is a matrix multiplication.

6. If the vector space is the set of functions of finite energy supported on 0-1, the endomorphism e is the linear functional we encountered when we studied graphon signals.

7. The object of interest that arises is the collection of all endomorphisms. This is a space we denote as End-of-M. If this sounds complicated it is because we are trying to be abstract and general (not supercilious!).

8. We are just referring to things like the space of all matrices. Or the space of all linear functionals.

Slide 7: The Set of Endomorphisms is a Vector Space

1. We now need to make an extra abstraction step and observe that the set End-of-M is also a vector space. This is the vector space of endomorphisms of M. Its definition requires that we define the elements of the vector space along with the sum and the scalar product operations. The elements of the set are the endomorphisms of M. All the linear maps from M onto itself.

2. The sum operation is the endomorphism e, which, when applied to x, yields the sum of the results of applying e_1 to x and e_2 to x.

3. And the scalar multiplication operator yields the endomorphism e-prime, which, when applied to x, yields the scaling of the result of applying e to x. Observe how in both cases we define operations in the vector space of endomorphisms in terms of operations in the vector space M. The sum of two endomorphisms is defined in terms of the sum of

two vectors. And the scalar multiplication of an endomorphism is defined in terms of the scalar multiplication of a vector.

4. It is important to remark that the vector space of endomorphisms is not the same as the vector space M. In the case of signals in R-n, the space of endomorphisms is the set of square matrices with n rows and n columns. In the case of finite energy signals in 0-1, the space endomorphisms is a space of linear functionals. Like all bounded functions in 0-1-squared.

Slide 8: The Set of Endomorphisms is an Algebra

1. The set of endomorphisms is not only different from the space M. It is also a set that has more structure. End-of-M is not just a vector space but it is also an associative algebra with unity. We signify that in the figure by changing its color.

2. To make End-of-M an algebra we need to endow it with a product. To do so we define the endomorphism e as the product of endomorphisms e_1 and e_2 if e is the composition of e_1 and e_2. The results of applying the two linear maps in tandem.

3. In the case of R-n, this is just the product of two matrices.

4. In the case of functions in 0-1 with finite energy, this is the composed functional given by a double integral.

5. Linear algebra is the processing of signals through the composition of linear maps. The study of the endomorphisms of vector spaces.

Slide 9: Signal Processing in the Algebra of Endomorphisms

1. Linear algebra is a form of signal processing. It is a form of processing in which any out of the set of all possible linear transformations is applied to input signals.

2. That we can apply **any** linear transformation, means there is no structure in the space of endomorphisms. This sentence can be a little vague, so let us be precise that what we mean here is that we were to learn in the space End-of-M, learning would not scale.

3. Introducing structure entails introducing a restriction in the set of allowable endomorphisms.

4. We know from experience that to accomplish this we have to introduce convolutional signal processing. We will do that with the introduction of algebras and representations in the following videos.

12.2 Algebraic Signal Processing

Slide 10: Algebra Signal Processing - Title Page

1. In the previous video we saw that the linear processing of a signal can be expressed as the application of an endomorphism on a vector space. We also highlighted that the space of all endomorphisms of a vector space is an algebra. One however that does not allow for the exploitation of signal structure.

2. We know that the introduction of convolutional filters is necessary to leverage structure. In this video we explain the use of algebras and homomorphisms to restrict the set of allowable linear transformations that can be applied to a signal.

Slide 11: From Linear Algebra to Signal Processing

1. The signals we want to process are elements of a vector space M. Objects like vectors, sequences or functions.

2. The linear processing of signals in M is undertaken by elements e End-of-M. The algebra of endomorphisms of M.

3. This is too general as a means of signal processing. We therefore want to restrict the set of allowable operations.

4. We will do that by restricting the endomorphisms *e* to the those that represent another, more restrictive, algebra.

5. That is, we introduce an algebra that defines the structure of the filters we want to apply to our signals x.

6. Filters are elements a of this algebra.

7. Which we map into the algebra of endomorphisms with the application of a homomorphism rho.

8. For this block diagram to make sense, we need to introduce and explain the notion of homomorphism and representation of an algebra.

Slide 12: Homomorphism Between Algebras

1. We begin with the definition of homomorphisms between algebras.

2. Consider then, algebras A and A'.

3. A homomorphism from A **to** A' is a map rho from A to A' that preserves the operations of A.

4. Namely for all elements a and b in the Algebra a:

5. The homomorphism preserves the sum. In the sense that applying rho to the sum of a and b is the same as summing rho-of-a and rho-of-b. Notice how the sums here are different. The sum of the algebra A is used on the left hand side. The sum of the algebra A' is used in the right hand side.

6. The analogous is true of the product. Namely, multiplying a and b in the algebra A and applying rho is the same as applying rho to a and b separately and multiplying them in the algebra A'.

7. The scalar product, is also preserved by the homomorphism.

8. The conditions in the definition of a homomorphism are such that carrying operations in the Algebra A is the same as carrying operations in the algebra A'. This is useful if, for example, the operations in A' are easier in some way. It is also useful if you are given an Algebra A and you want to process signals in a vector space M.

9. Do notice that converse need not be true. Operations in A and A' are not equivalent. We can move from A to A' but it is not necessarily true that operations in A' can be mapped to operations in A. This could happen because A' has more elements than A.

Slide 13: Representations and Filters

1. From the definition of homomorphism, flows the concept of a representation.

2. Given an associative Algebra A and a vector space M, we consider a homomorphism rho from A to End(M). That is, an operation-preserving map from the algebra A to the space of endomorphisms of M.

3. The pair M-rho, made up of the vector space M and the homomorphism rho is said to be a representation of the associative algebra A.

4. The representation ties the abstract algebra A to concrete operations on signals that live in the vector space M.

5. For this reason we henceforth say that elements a of A are filters. The action of the filter a on the signal x produces the filtered signals rho-of-a times x. That is the action of the filter a on a signal x is the application of the endomorphism rho-of-a.

Slide 14: Algebra of Polynomials of a Single Variable

1. All of this looks very complicated because it is very abstract. But it is actually quite simple. To illustrate concepts we consider the algebra of polynomials and explain how graph signal processing is recovered as a particular case of algebraic signal processing.

2. A polynomial over a field F is an **expression** having the familiar look of a polynomial. We have a sum of powers of t modulated by coefficients a_k.

3. But the interpretation of this familiar expression is a little different. The coefficients here are elements of a field. This is as usual. But the sum and the powers of t are just symbols. They do not represent an actual sum or an actual power of a variable. They are just scribbles in a piece of paper.

4. The algebra of polynomials over F is the the set of all the polynomials taking coefficients in F along with the following operations:

5. The scalar multiplication of a by alpha is **defined** as the multiplication of the polynomial coefficients a_k. This multiplication is **not just a symbol**. It amounts to the product of two elements of the field. This is a product of real numbers if we are considering the algebra of polynomials over the reals.

6. The vector sum of two polynomials is defined as the polynomial whose coefficients are the sum of the given polynomial coefficients. Again, the sum highlighted in blue here is

an actual sum. The sum of elements of the field. The sum of two real numbers, for instance.

7. Finally, the algebra product is defined as a polynomial in which the coefficient multiplying t-k is the sum of the products of coefficients a_j and b_k-minus-j. In each of these products the sum of the a-subindex and the b-subindex is k. Again, the operations highlighted in blue are operations in the field. The interpretation of this algebra is that we have abstract symbols we call polynomials. These abstract symbols can be manipulated with these three operations. And these three operations satisfy the conditions stated in the definition of an algebra.

Slide 15:Graph Signal Processing

1. Our interest in the algebra of polynomials is that we can use it to generate several known instances of signal processing. We can, in particular, use it to generate graph signal processing.

2. To see how this is done, consider signals x that live in the vector space R-n. The space of endomorphisms to process these signals is made up of all the square matrices E with n rows and n columns.

3. The challenge is that processing x with an arbitrary E in this space of endomorphisms is too general. We need some extra structure that we can exploit. One possibility is to assume that the signal x is supported on a graph with shift operator S.

4. To leverage this information we define the homomorphism rho that maps the polynomial a to the polynomial rho-of-a in which the **symbol** t to the power of k is replaced by the **matrix** S to the power of k. These two polynomials look similar but they are objects of a different nature. The polynomial on the left is just a symbolic expression. The polynomial on the right is a matrix in End-of-M. We use the operations of the algebra of endomorphisms to sum powers of S modulated by coefficients h_k. Being a concrete matrix, we can apply it to a given signal x. We have **instantiated** the abstract filter *a* into a concrete representation that we can apply to our signals of interest.

5. The polynomial rho-of-a is the definition of a graph filter. We can therefore say that the combination of the algebra of polynomials with the homomorphism rho we defined here yields graph signal processing on the graph shift operator S. The value of the abstract algebraic formulations is that several other versions of signal processing can be recovered from different choices of algebras and homomorphisms.

Slide 16: Algebraic Signal Processing (ASP)

1. We can now reintroduce our block diagram defining algebraic signal processing models. An ASP model is a triplet A-comma-M-comma-rho.

2. In the ASP model A is an algebra with unity where filters h are defined.

3. The algebra defines the **rules** of convolutional signal processing. It is defined in the abstract. At the symbolic level of squiggles in a piece of paper.

4. The second component of the ASP model is a vector space M.

5. This is the vector space containing the signals that we want to process. This is, typically, a vector space where we can write objects made up of numbers that we can add and multiply according to some rules.

6. And the third component, rho, is a homomorphism from the abstract algebra A to the algebra of endomorphisms of M.

7. It instantiates the abstract filter h in the space End(M). It makes the filter into a linear transformation that we can apply to the signals x that are given to us.

8. Any filter h in the algebra A is a filter that can operate on the signals x as dictated by the homomorphism. The result of applying h to a signal x is the linear transformation rho-of-a applied to the signal x. We have seen that **graph** SP is a particular case of **algebraic** SP. We will see that several other **whatever-**SPs are also possible. The value of **algebraic** SP is that it provides a common framework for joint investigation of their shared fundamental properties.

12.3 Polynomials in an Algebra and Polynomial Functions

Slide 17: Polynomials in an Algebra and Polynomial Functions - Title Page

1. Polynomials and Polynomial functions play a central role in algebraic signal processing. This section is a short aside to introduce definitions that we will use later on.

Slide 18: Polynomials in an Algebra

1. Things are about to get insidious We are going to have several different objects, all of which are called polynomials. But their subtle differences are precisely what is important to be able to understand the foundational concepts of algebraic signal processing.

2. Our first polynomial is a polynomial in an algebra. Given an element a of A and a set of coefficients h_k in the field F, a polynomial is another **element** of the algebra.

3. We denote this element as p_A-of-a and define it through the use of the usual polynomial expression.

4. This involves several terms in each of which have the algebra element *a* multiplied by itself k times.

5. And the result further multiplied by the coefficient h_k.

6. These different terms are then added together. This is what you should had expected for an expression that is called a polynomial. There is nothing new here.

7. Nevertheless, there are two important points to emphasize about this polynomial. The first is that p_A-of-a is an element of the algebra A. We know that this is true because p_A-of-a is constructed from a by using the operations of the algebra.

8. The second point is precisely this. That p_A-of-a is generated using the operations of the algebra. The polynomial is **not just a symbol,** as was the case when we defined the algebra of polynomials. It is a **concrete** representation of a **concrete set of operations.** This distinction, already subtle, is made even more subtle because the operations that generate p_A-of-a can be symbolic. In any event, symbolic or not, the polynomial p_A-of-a is an element of the algebra A that we obtain through a concrete set of operations that we perform on the algebra element a.

Slide 19: Polynomials in a Different Algebra

1. The reason we want to emphasize the use of the algebra's operations is that we are going to be interested in writing polynomials on different algebras. When we change the algebra, it is possible that we use the same coefficients. But, although this may result in two polynomials that **look the same,** the polynomials are actually different.

2. To drive this point suppose that we consider an element a-prime of a different algebra A-prime. Even if we retain the same coefficients h_k the polynomial p_A-prime-of-a-prime is a different element.

3. This is not only because a and a-prime are different. But also because we are using a different set of operations. We are now using the operations of the algebra A'.

4. Notice that this can get obscured with the usual notation because it is implicit that the operations are performed in the right algebra. The important point to keep in mind is that polynomial expressions that look the same, can represent different operations. Like operations on an algebra A. And operations on a space of endomorphisms of a vector space. As you may foresee.

Slide 20: Polynomial Functions over a Field

1. As if we didn't have enough polynomials already, we have to introduce yet another concept. This is that of a polynomial function over a field. This is notoriously different from the others in that it does not involve an algebra.

2. We consider coefficients over a field along with a variable lambda that takes values on the same field.

3. The polynomial function p_F, is the function that maps lambda to the value p_F-of-lambda defined by an expression with the usual polynomial terms.

4. In each of these terms we have lambda multiplied by itself k times.

5. With the result being further multiplied by the coefficient h_k.

6. And the different terms added together to obtain the polynomial's evaluation.

7. This polynomial is the one that should be most familiar. It is just a function of lambda. Mapping values on the field to other values on the field. It is, we must repeat, a function that is defined in terms of the operations of the field. There is no algebra involved here.

8. By the way, if you notice a resemblance to frequency responses, this is not coincidental. In fact, it is rather the point. But this is just an aside comment at this point.

Slide 21: Polynomials with Multiple Elements in a Commutative Algebra

1. In our discussions so far we have considered polynomials of a single variable. Generalizations to polynomials with multiple elements are ready. If we consider a set calligraphic A made up of r element of the algebra A, we define the polynomial p_A-of-A:

2. As the summation of different terms.

3. Each of which has the different elements of calligraphic A raised to different powers.

4. And scaled by corresponding coefficients.

5. This polynomial is associated with coefficients h-sub-k_1-through-k_r that are drawn from the field F. This polynomial represent a concrete set of operations performed on the elements of the set calligraphic A to obtain another element of the algebra. The polynomial p_A-of-A. We point out that in this definition the algebra is assumed to be commutative. If the algebra is **not** commutative, the order of the products matters and the polynomial takes on a different form. This is not unimportant. There are interesting algebras that are not commutative. But the extension is straightforward.

6. Likewise, we can define the corresponding polynomial function with multiple variables lambda_1 through lambda_r. If we use the same set of coefficients, this polynomial function is the same expression as before in which the different algebra elements a_i are replaced by variables lambda_i.

7. Nevertheless, the similarity of the expressions should not distract from the fact that the objects they represent are very different.

8. This second expression is a function of the variables lambda_i. It takes different values when we instantiate different variables. It is also an expression that uses different operations. The polynomial on a_i uses the operations of the algebra A. The polynomial on lambda_i uses the operations of the field.

12.4 Generators, Shift Operators, & Frequency Representations

Slide 22: Generators, Shift Operators, and Frequency Representations - Title Page

1. Different forms of convolutional signal processing can be recast into the common abstract framework of algebraic signal processing. We use algebras and homomorphisms to define different types of convolutional filters.

2. In the analysis of these filters there are three central components that appear: Generators, Shift Operators, and Frequency Representations. We cover these three concepts in this section

Slide 23: Generators of an Algebra

1. Generators, shift operators and frequency response are intimately related to polynomials. The first appearance of polynomials is in the definition of generators and generator sets.

2. Indeed, we say that the set G generates the algebra A if all the elements h of A can be represented as polynomials on the elements of G.

3. That is, any H in A can be written,

4. As a sum of terms involving different powers of the generators g modulated by certain coefficients.

5. We use p_A-of-G to denote the polynomial that generates h from g. As per the definition this polynomial is just the algebra element h. The notation p_A-of-G is just meant to emphasize that it is a polynomial.

6. We will say that the elements g of G are generators of A and we will say that p_A-of-G is the polynomial that generates h.

7. The meaning of this definition is that filters h can be build from the generating set using the operations of the algebra.

8. When this is possible, we end up with a remarkable property: When given the algebra, the generators are given. Thus, a filter h is completely specified by its coefficients. If we want to understand the effect of a filter, we don't necessarily have to look at the filter itself. It may suffice to look at its coefficients.

Slide 24: Generators of the Algebras of Polynomials

1. One may wonder if interesting generator sets exist at all. They do in a variety of algebras. One particular case where this is rather obvious is the algebra of polynomials of a single variable. This algebra can be generated by the polynomial g equals t.

2. To see that this is true recall that algebra elements are symbolic expressions where we write sums of powers of t modulated by coefficients h_k.

3. These symbolic expressions can be generated from g equals t with a polynomial where we write sums of powers of t modulated by coefficients h_k.

4. OK. Let's rewind. The difference here is difficult to see.

5. In this expression the sum, the letter t and the powers of k are just symbols. They are not representing actual operations. They are just squiggles in a piece of paper that we use to represent an element of the algebra of polynomials.

6. In this other expression, t is denoting a particular element of the algebra of polynomials. And the powers of k and the sum are representing actual operations: The operations of the algebra of polynomials.

7. If we apply the operations symbolized here to the polynomial g equals t, we generate the filter h. Since this is possible for all filters, it follows that the element t generates the algebra of polynomials.

8. A similar generation is possible if we consider the algebra of polynomials of two variables x and y. This algebra is one that we can generate with the polynomials g_1-equals-x and g_2-equals-y.

9. That this is true follows because the algebra of polynomials of two variables is defined by the expected polynomial expression.

10. Which we can alternatively rewrite using the operations of the algebra on the polynomials g_1-equals-x and g_2-equals-y.

11. As in the case of single variable polynomials we need to rewind here.

12. This is just an expression. There are no actual operations being performed when we encounter sums products and powers

13. This is a representation of operations. We are applying the operations of the algebra of polynomials of two variables to the polynomials x and y. By the way, we have not

formally defined what these operations are. But they are straightforward extensions of the definitions we gave for single variable polynomials.

14. (Empty)

Slide 25: Shift Operators

1. When given generators along with a representation, we define shift operators.
2. Formally, let M-comma-rho be a representation of an Algebra A.
3. And calligraphic G be a generator set of the algebra A.
4. We say that S is a shift operator.
5. If we can write S as rho-of-g for some element of the generator set. The shift operators are the images of the generators when applying the homomorphism rho.
6. The collection of all shift operators is the set calligraphic S. This is the set of shift operators of the **representation** M-rho of the **algebra** A. Both, the representation and the algebra determine the elements of the set of shift operators.
7. Recall that the homomorphism is a map from the algebra to the space of endomorphisms of the vector space M. Thus, the generators g of the algebra are mapped to the shift operators S in the space End-of-M. The abstract generator is mapped to a a concrete linear transformation that we can apply to a signal x.

Slide 26: Polynomials on Shift Operators

1. The introduction of shift operators brings us to the second polynomial of the section. This one appears in the form of a theorem stating that filters h can be represented in the space of endomorphisms as polynomials on the shift operators.
2. Formally and more precisely, let M-rho be a representation of the Algebra A.
3. And further consider a set of generators g_i.
4. Along with a set of shift operators S_i.

5. We then claim that the representation rho-of-h of the filter h is a polynomial on the shift operator set.

6. That is, if we write the filter h as a polynomial p_A-of-G

7. We can write the homomorphism image rho-of-h as a polynomial p_M-of-S.

8. Both of these polynomials have the same coefficients. But they use different variables.

9. The polynomial p_A-of-G that defines the filter, uses the generators as variables.

10. The polynomial p_A-of-M that instantiates the filter in the space of endomorphisms uses the shift operators as variables.

11. These polynomials also differ in that the operations involved are different. The polynomial p_A-of-G that defines the filter uses the operations of the algebra A. The polynomial p_A-of-M that instantiates the filter in the space of endomorphisms uses the operations in the space of endomorphisms of M.

12. (Empty)

13. To prove the theorem we simply recall that the homomorphism preserves the operations of the algebra.

14. Indeed begin by writing the image rho-of-h

15. As the application of rho to the polynomial p_A-of-G.

16. But since operations are preserved by the homomorphism, it doesn't matter if we apply the operations before or after the application of the homomorphism.

17. Noticing that rho-of-g_i equals S_i concludes the proof of the theorem.

Slide 27: Algebraic Signal Processing (ASP)

1. Generators and shift operators look interesting. But are they useful? They are **very** useful. The theory of algebraic signal processing involves the specification of:

2. A vector space of signals.

3. Which comes with an associated algebra of endomorphisms. A set that contains all the linear transformations we can apply to a signal x.

4. Separate from the space of signals. And it is important to remark that this is **completely separate** from the space of signals. We have an algebra A that specifies the filters of interest.

5. Allowable filters are elements h of the algebra A,

6. To tie filters and signals we have the homomorphism rho. This is the instantiation of the filter in the space of endomorphisms of M. It makes the abstract filter into a linear operator rho-of-h.

7. This is a linear operator that we can apply to the signal x to produce the filtered signal rho-of-h-x.

8. The hitch is that, in principle, the homomorphism has to be specified for all filters rho.

9. Given filter h_1 we have to specify the image rho-of-h_1

10. Given filter h_2 we have to specify the image rho-of-h_2

11. For filter h_3 the image rho-of-h_3. And so on. Since all of these is an abstract analytical tool, this is not necessarily a concern. But it doesn't bode well for our ability to analyze ASP models.

12. If we have a generating set, however, it suffices for us to specify the homomorphism rho for the elements of the generator set.

13. Given g_1 we specify the image rho-of-g_1.

14. Given g_2 we specify the image rho of g_2. If we have two generators, this is it.

15. Because all other filters can be written as polynomials on the generators.

16. Which we know instantiate to polynomials on the shift operators. The polynomials on g_i and S_i are the same in that they use the same coefficients. They are different in that they rely on different operations. Most importantly, they are homomorphic. The map rho applied to the polynomial p_A-of-G yields the polynomial p_M-of-S.

Slide 28: Graph Signal Processing

1. To illustrate this simplification we can consider graph signal processing, We have already seen that we can recover GSP from ASP if we consider signals in R-n. Along with the algebra of polynomials. ANd a homomorphism rho in which the polynomial h on the abstract symbol t is mapped to a polynomial on the shift operator S.

2. This specification of the homomorphism is equivalent to the much simpler specification in which we map rho-of-t to S.

3. These specifications are equivalent because we know that the algebra of polynomials is generated by g-equals-t.

Slide 29: Frequency Representation of an Algebraic Filter

1. The third and final polynomial of the section is the frequency representation of an algebraic filter.

2. To define this formally we consider an algebra A with generators g_i. In this algebra we are given a filter h expressed in its polynomial form p_A-of-G.

3. The frequency representation of the filter h over the field F is the polynomial function p_F-of-L with variables lambda_i. The polynomials p_A-of-G and p_F-of-L use the same set of coefficients. The expressions differ in that p_A-of-G involves powers of the **generators** g_i whereas p_F-of-L involves powers of the **variables** lambda_i.

4. We do remark that the similarity of the expressions should not obscure the fact that these two polynomials are creatures of very different natures. The polynomial p_A-of-G uses the operations of the algebra to build a filter h. The polynomial is itself an element of the algebra. The frequency representation is a function that relies on the operations of the field. Often, the frequency representation is a simpler object

Slide 30: Three Polynomials (or More)

1. We have seen the introduction of the three central components of ASP models. The three of them are polynomials:

2. The filter

3. The instantiation of the filter in the space of endomorphisms.

4. And the frequency response.

5. These three polynomials are insidiously similar because they have the same coefficients. Their expressions look about the same. Yet, although they are related. They are also quite different.

6. They are objects of different natures. They live in different spaces. They utilize different operations. And they have different meanings. Let us review these differences.

Slide 31: Polynomial 1: The Filter

1. Polynomial number 1 is the filter.

2. This is the polynomial p_A-of-G. It is a polynomial on the algebra elements g_i. Which are the generator elements of the algebra A.

3. In the definition of this polynomial we use the operations of the algebra. The sum, the product, and the scalar product of the algebra.

4. This polynomial is an abstract definition of a filter. It is **untethered** to any specific signal model.

Slide 32: Polynomial 2 (or More): The Instantiation of the Filter

1. Polynomial number 2 is the instantiation of the filter in the space of endomorphisms of the vector space M where the signals x live.

2. This is the polynomial p_M-of-S. It is a polynomial on the shift operators S_i. Which are the images of the generator elements of the algebra A.

3. In the definition of this polynomial we use the operations of the algebra of endomorphisms. The sum, the product, and the scalar product of the algebra End-of-M.

4. This polynomial is a concrete instantiation of the abstract filter. It determines the concrete effect that a filter has on a signal x. The instantiation of the filter is **tethered** to a specific signal model

5. This is also where the mysterious “or more” of the last few slides can be explained. The same abstract filter can be instantiated in multiple signal models. This is a powerful realization. It has been at the core of our stability and transferability analyses.

Slide 33: Polynomial 3: The Frequency Response

1. Polynomial number 3 is the frequency response.

2. This is the polynomial p_F-of-L. It is a polynomial **function** with variables lambda. We have as many lambda variables as generators the algebra has.

3. In the definition of this polynomial we use the operations of the field. There are no three operations as in the other two polynomials. Only two. The field product and the field sum. This is also not the processing of an element of an algebra to produce another element of the algebra. This is a function that takes as input elements of the field and produces a different element of the field. This is something like a polynomial with real variables.

4. In that sense, this is a simpler representation of a filter. It is a representation that is also untethered to a specific signal model. Except for the field. Which has to be the field over which the vector space M is defined.

5. The frequency response is the tool we use to analyze filters. The one we have used in GSP to explain discriminability, stability, and transferability. The one on which we are going to derive more general results.

Slide 34: The Three (or More) Polynomials in GSP

1. In the concrete case of GSP the abstract filter is an element of the algebra of polynomials of a single variable. This is an abstract definition of a graph filter. It is untethered to any specific graph.

2. The filter instantiation entails mapping the generator t of the algebra of polynomials to the shift operator S of the graph. We therefore instantiate the abstract polynomial into a polynomial on the shift operator S. This is what we have called a graph filter. It is tethered to a specific graph.

3. A very interesting point to note is that the abstract graph filter can be tethered to a different graph. To do that, we just need to map the generator t of the algebra of polynomials to the shift operator S-hat of this other graph.

4. We don't even need to stay within the space of graphs. We can tether the filter to a graphon. We do that by mapping the generator t of the algebra of polynomials to the graphon shift operator W.

5. The frequency response in this case is a polynomial on a single real variable. This, as it should, coincides with our earlier definition of the frequency response of a graph filter. An important property that it has, is that the frequency response is a property of the abstract filter h. The frequency response is not tethered to a specific instantiation of the filter. It is the same for all graphs or graphons.

12.5 Convolutional Information Processing

Slide 35: Convolutional Information Processing - Title Page

1. Algebraic filters provide a generic framework out of which we can extract the commonalities of different forms of convolutional information processing.

2. To substantiate this claim we have to show that, indeed, it is possible to express familiar convolutional filters in the language of algebraic filters. Doing so requires the specification of vector spaces, algebras, and homomorphisms. Which we do in this section for graph, time, and image processing.

Slide 36: Specification of ASP Models

1. In order to specify a concrete ASP model, we need to prescribe three objects:

2. A vector space M where signals x live.

3. The prescription of this vector space implicitly prescribes a space of endomorphisms. Which is the set of linear transformations that can be applied to signal x.

4. **Separate** from the vector space M, we prescribe and Algebra A where filters h live. This algebra defines the rules of convolutional processing in an abstract sense.

5. The third element we prescribe is a homomorphism rho that maps filters to linear transformations we can apply to signals x. The homomorphism translates the abstract filter h to a concrete filter rho-of-h we can apply to signals.

6. Very importantly, the specification of the homomorphism can be reduced to the specification of shift operators. These are the images rho-of-g of the algebra generators g

7. If we specify the mapping of the generators, the homomorphism gets defined for any other filter. This is because filters h are polynomials on the generators g which are mapped to polynomials on the shift operators S with the same coefficients.

8. The prescription of M, A, and rho determines the convolutional processing of signals x as the application of the endomorphism rho-of-h. This allows the use of algebraic models to solve a variety of signal processing tasks. Including Graph, Graphon Time, and Image processing as we describe next

Slide 37: Graph Signal Processing (GSP)

1. We begin by reviewing graph convolutions. Which we have already seen can be expressed as particular forms of algebraic convolutions. More to the point, we begin by tackling a graph signal processing task where our goal is to process a signal x that is supported in a graph. The graph has n nodes supporting each of the n entries of the signal x. And the graph is described by a shift operator S.

Slide 38: Graph Signal Processing (GSP)

1. GSP in the graph with shift operator S is a particular case of ASP in which:

2. The vector space where signals live is R-n

3. The associated space of endomorphisms is the space of square matrices

4. The algebra is the algebra of polynomials of a single variable. In which filters are written as symbolic polynomials

5. The mapping to the space of endomorphisms is accomplished by mapping the generator t of the algebra of polynomials to the shift operator S of the graph.

6. This results in filters expressed as polynomials in the shift operator S. Which is, indeed, the object we have called a graph fitler.

7. The result of applying this filter to a signal x is simply the multiplication of the polynomial on S with the signal x.

Slide 39: Graphon Signal Processing (WSP)

1. Another task we can solve with algebraic filters is the processing of signals on graphons. The signals that we are given are supported on the interval zero-one and have finite energy. They belong to the space L_2 of 0-1. Which contains the functions that are square integrable in this interval.

Slide 40: Graphon Signal Processing (WSP)

1. To process signals on graphons we particularize ASP so that:

2. The vector space where signals live is L_2 of 0-1. The space of functions that are square integrable in 0-1.

3. We still stick to using the algebra of polynomials of a single variable. Filters, in the abstract, are the same as the filters we use in **graph** signal processing.

4. What changes is the definition of the homomorphism. In the case of **graphon** signal processing we map the generator of the polynomial algebra to the operator T_W. This is the integral function operator that applies the graphon W-of-u-v to the graphon signal x-of-v.

5. This mapping completely defines the homomorphism and results in filters that are given as polynomials on the operator T_w. Where powers of the operator T_W represent successive applications of the operator.

6. The application of a graphon filter to a signal x is the application of this polynomial operator to input signals. Which is, indeed, the structure we have called a graphon filter.

Slide 41: Discrete Time Signal Processing (DTSP)

1. We can now move on to consider the processing of discrete time signals. These are sequences X defined over the integers. They extend from minus to plus infinity. The sequences are required to have finite energy. They are square summable. We say they live in L_2-of-Z.

Slide 42: Discrete Time Signal Processing (DTSP)

1. Discrete time signal processing is also a particular case of ASP. In this case we have that

2. The vector space where signals live is L_2-of-Z. The space of square summable sequences.

3. The algebra is, again, the algebra of polynomials of a single variable. Same as in graph and graphon signal processing.

4. What changes is, again, the homomorphism. We produce a homomorphism in which the polynomial t that generates the polynomial algebra is mapped to the **time** shift operator S. The time shift operator is a linear operator that, when applied to the sequence X, it shifts the time indexes one unit up. The new sequence has the same values. But the value at index n after application of the operator, is the one associated with the index n-1 in the sequence to which the operator is applied.

5. This mapping of the generator t, yields filter instantiations in the form of polynomials on the **time** shift operator.

6. When we apply this filter we produce an expression that is not unlike the ones we've seen before. Except that the shift operator S is now a **time** shift operator. Not a graph or graphon shift operator. An extra level of simplification is possible if we isolate the n-th component of the output. This can be written as the sum over k of h_k times X-sub-n-minus-k. Which is the familiar expression for the convolution of time signals.

Slide 43: Image Processing (IP)

1. Yet another example of a task that we can solve with **algebraic** SP is the processing of images. An image is mathematically represented as a sequence with two integer

indexes. The sequence is required to have finite energy, as usual. We say that it belongs to L_2-of-Z-square.

Slide 44: Image Processing (IP)

1. To recover image processing as a particular case of algebraic signal processing we define:

2. M as the vector space of square summable sequences with two indexes. The space L_2-of-Z-square.

3. In this case we need to use a different algebra: The algebra of polynomials of two variables. Made up of abstract expressions we can write as polynomials with two letters.

4. This is an algebra that has two generators. The polynomial x and the polynomial y. We map the polynomial x to the shift operator S_x. This is an operator that shifts the **first** coordinate of the two-index sequence. We map the polynomial y to the shift operator S_y. This is an operator that shifts the **second** coordinate of the two-index sequence.

5. This choice of mapping of the generators x and y yields filters that are expressed as polynomials that signify repeated applications of the shift operators S_x and S_y.

6. Applying these polynomial operators to the processing of an image X produces an expression that is equivalent to the usual definition of the 2-dimensional convolution that is used in image processing.

Slide 45: ASP is a Generic Analysis Tool

1. Algebraic convolutional filters encompass graph, graphon, time, and image convolutional filters as particular cases.

2. There are other particular cases that can be obtained/. Notably, group convolutional filters.

3. The value of having built this abstraction is that it provides a framework for analyses that hold for all kinds of convolutional filters. Our particular interest will be in constructing an abstract framework for algebraic convolutional neural networks. Out of these framework we expect to find abstract stability properties that hold for all particular cases. Namely, graph, graphon, time, image and group convolutional neural networks.

12.6 Algebraic Neural Networks

Slide 1: Algebraic Neural Networks

1. With the concept of algebraic filtering in place, we will introduce the algebraic neural network architecture in this video.

Slide 2: Algebraic Neural Networks

1. Algebraic neural networks (AlgNNs) are defined as a stacked layered structure.
2. Each layer consists of the algebraic signal model (A_l, M_l , rho_l) and the map sigma_l that is a composition of a nonlinearity operator and a pooling operator. Like in any ASP model the pair M_l and rho_l is a representation of A_l. The mapping between layers is performed by sigma_l.
3. At layer l, the input x_(l minus 1) is first processed by the filter rho_l of a_l for the intermediate feature
4. Then, the intermediate feature passes through the nonlinearity and pooling sigma_l to generate the output x_l.
5. The above operation can be represented equivalently as a nonlinear map Phi of x_(l minus 1),calligraphic P_l , calligraphy S_l
6. Where calligraphic P_l is contained in A_l and highlights the properties of the filters and calligraphic S_l is the set of shifts associated to the representation pair M_l and rho_l

Slide 3: Algebraic Neural Networks

1. This figure shows the architecture of the Algebraic Neural Network with three layers. At each layer, there is an algebraic signal model. We first process the input with the algebraic filter, then process the result with the nonlinearity and the pooling, and then generate the output for the next layer.

Slide 4: Convolutional Features

1. In fact, the processing at each layer can be performed by a family of filters, which leads to multiple features.

2. To be more precise, the F_(l minus 1) inputs are processed by F_l F_(l minus 1) filters rho_l of a_l^gf to generate intermediate features. The latter are first aggregated over the input index g and then passed through the nonlinearity and the pooling sigma_l to output features at layer l.

3. Here, a_l^gf is the filter processing gth feature x_(l minus 1)^g and F_l is the number of feature at layer l.

4. Note that layers may use different and specific algebraic signal models (A_l, M_l, rho_l). In particular, the algebras can be different but they are usually chosen as the same.

5. The trainable parameters are the filters a_l^gf, while numerically, we directly train on rho_l of a_l^fg.

Slide 5: Pooling

1. The pooling operation is meant to increase the computational efficiency and to improve the performance.

2. In a general Algebraic Neural Network this operation is attributed to the operator sigma_l which is the composition of the operators eta_l and P_l.

3. Where eta_l is a pointwise nonlinearity and P_l is a pooling operator.

4. The only property assumed from sigma_l is to be Lipschitz and to have zero as a fixed point, this is sigma_l of zero equals zero.

5. It is also important to point out that the pooling operator P_l projects elements from a given vector space M_l into another M_(l plus 1).

Slide 6: Example 1: CNN

1. We now provide explicit examples for the Algebraic Neural Networks. We first consider traditional CNNs, which particularize the algebraic signal model in AlgNNs as the typical signal model.

2. Make M equals C^N and A the polynomial algebra in the variable t and module t power N minus 1.

3. And S equals C is the cyclic shift operator satisfying C power k equals C power k module N.

4. Substituting above definitions then yields the operation of the traditional CNN layer.

5. In this case, the pooling operator P_l is a sampling operator while the nonlinearity eta_l is the ReLU.

Slide 7: Example 2: GNN

1. The second example is the Graph Neural Networks, which particularize the generic algebraic signal model to that one of graph signal processing

2. Let the vector space M equals C^N with the nth component x_n of the signal x in M associated with the nth graph node.

3. Let also A be the polynomial algebra in the variable t

4. The homomorphism filter rho of a is then given by the polynomial in the shift S.

5. Here, S is the graph matrix representation, such as the adjacency matrix, the Laplacian matrix and so on.

6. Substituting these definitions, we can obtain the operation at each GNN layer.

12.7 Perturbation Models

Slide 1: Perturbation Models - Title Slide

2. In this lecture we introduce the notion of perturbation for generic algebraic signal models

Slide 2: Representation and Shift Operators

2. In order to properly define the model deformations we first recall the notion of generator of an algebra

3. We say that the set calligraphic G is a generator of A, if every element of the set A can be built from calligraphic G as polynomials using the operations of the algebra. In other words, the elements of A can be generated as a polynomial of the elements of the set calligraphic G

4. The Shift operators are the set of images of the homomorphism applied to the generators.

5. In short, the generator set of A is the set with which we can reconstruct A by using polynomial functions. And we define the shift operators as the set of all the results of applying the homomorphism to the generator set

6. Then, we represent the filter \rho of a as a polynomial function of the set calligraphic S, which is the set of shift operators

Slide 3: Perturbations in Algebraic Signal Models

6. We define perturbations of Algebraic signal models as perturbations of the shift operators

7. Then, the perturbed model is associated to S tilde which equals S plus a perturbation function T of S

8. The algebraic signal processing model given by A coma M coma rho is consequently perturbed to the triplet (A coma M coma rho tilde) such that

9. Rho tilde of a is equal to P_M of rho tilda of the generator g, which equals the polynomial function p_M of calligraphic S tilde

10. That is, the polynomials that define filters are the same. But they use the perturbed shift operator

11. As an example let's think about graphs. The shift operator S represents the graph and the perturbed model S tilde represents another graph, which is a modified version of the graph associated to the operator S

12. Or in the context of time signals, if the shift operator S represents translation equivariance, S tilde represents the quasi-translation equivariance property, which is obtained as a perturbation of the operator S

Slide 4: Perturbation Definition: Remarks

6. Our definition of perturbation limits the perturbation of the homomorphism to the perturbation of the shift operator.

7. This is a somewhat arbitrary choice but one that is motivated by practice as we illustrate for graph signals, discrete time signals, signals on groups, graphon signals

8. The model is also arbitrary in that it perturbs the homomorphism ρ but not the Algebra A or the signal x. This is also justifiable. Notice, first that the algebra A is a choice of operations and therefore not naturally subject to perturbation.

9. We can interpret a perturbation x tilde of the signal as a transformation of the shift S

10. This model is particularly useful as we can study the effect of perturbation T on signal x by processing the original signal x with the perturbed shift operator S tilde

11. This is, the effect of the perturbation T of x can be studied by processing x with the perturbed shift operator S tilde equals the composition of the shift operator S and T

Slide 5: Perturbation Model (Deformations)

6. The model of perturbation that we will study hereafter is the first order generic model of a small perturbation for the shift operator

7. Our model consists of two parts,

8. the absolute perturbation T_0

9. and the relative perturbation T_1 acting on the shift operator S

10. For the context of our analysis, we will only consider the operators T_0 and T_1 to be compact normal, and we will impose that the norm of both operators is bounded by one

11. Notice that this condition is not a strong requirement as we are interested in perturbations T of S whose norm is significantly smaller than one

Slide 6: Perturbation of Graph Filters

7. To illustrate the ideas discussed previously, we discuss the implications of the perturbation model considered for graph signal processing. In the case of graphs, to consider a perturbation means to apply the same filter, on the exact same signal. The only thing that changes will be the graph

8. If the original graph represented by the shift operator S is used, and a filter with coefficients h_k is considered, We process X instantiated on S as H(S) times X.

9. Then, when a perturbed GSP model is considered, we use the same filter coefficients h_k to process the same signal but instantiated on the perturbed graph shift operator S tilde

10. The perturbed model S tilde is then defined by the additive perturbation matrix T_0 plus the relative perturbation matrix T_1 one applied to the graph shift S

11. If you find this slide familiar, it is because we already used it. This slide was part of lecture 6.

12.8 Stability Theorems

Slide 1: Stability Theorems - Title Page

13. In this part of the lecture, we will define stability in the context of algebraic signal processing

14. And use this definition to show that algebraic filters and algebraic neural networks are stable.

Slide 2: Stability-recalling basic notions and notation

15. We recall that in the algebraic signal model (A coma M coma \rho), filters are defined by the operators acting on M that are obtained as the images of A under \rho

16. If A is generated by the set calligraphic G, any element of the algebra can be written as a polynomial function of the elements in the generator set

17. Consequently, any filter acting on the vector space M is defined by operators that can be expressed as functions of the shift operators

18. More specifically filters can be written as polynomial functions of the shift operator S that we denote as p(S)

Slide 3: Stability

1. Recall that stability is a property of an operator.

2. In the context of algebraic signal processing, we say that an operator p of S is stable

3. If there exist positive constants C sub 0 and C sub 1 such that, if S is perturbed by T to yield S tilde,

4. The norm of the difference between the operator p of S applied to an algebraic signal x, and the operator p of S tilde applied to the same signal x, is upper bounded

5. For all x.

6. In this definition, D sub T of S denotes the Frechet derivative of the perturbation T. As such, we see that the norm of the difference between the operators is bounded by the size of the perturbation in two ways. First, it is bounded by the value of the perturbation T itself, scaled by C sub 0. Second, it is bounded by the rate of change of this perturbation, which is given by D sub T of S and scaled by C sub 1.

7. Importantly, note that stability of algebraic operators is not a given. There are several examples of unstable operators in the processing of both graph and time signals. Which is why we analyze the stability of algebraic filters and neural networks.

Slide 4: Lipschitz and Integral Lipschitz Filters

1. Filters are polynomials on the shift operator.

2. Therefore, they are isomorphic to polynomials with complex variables. Which allows us to define filters as mappings from the complex numbers to the complex numbers.

3. A Lipschitz filter is defined as a Lipschitz polynomial p of lambda. That is, given two complex numbers \lambda and \mu, the polynomial p is such that the magnitude of the difference between p(\lambda) and p(\mu) is bounded by the difference between \lambda and \mu scaled by some constant L_0. L_0 is called the Lipschitz constant of the filter.

4. An integral Lipschitz filter is defined as an integral Lipschitz polynomial p of lambda. That is, the norm of the product of the derivative of p of lambda by lambda is bounded by some constant L_1. L_1 is called the integral Lipschitz constant of the filter. In integral Lipschitz filters the rate of variation approaches zero as lambda grows

5. To make things simple, in a first moment we restrict attention to algebras with a single generator. Generalizations to algebras with multiple generators are cumbersome but easy to obtain.

Slide 5: Stability of Algebraic Filters

1. We start by analyzing the stability of algebraic filters.

2. If an algebraic filter is Lipschitz and integral Lipschitz, it is stable, that is

3. The difference between the filter outputs on the shift operator S and on the perturbed shift operator S tilde is upper bounded by the size of the perturbation and by its rate of variation. In the case of the algebraic filter, the constants C sub 0 and C sub 1, which scale the norm of T and the norm of its Frechet derivative respectively, are given by the Lipschitz constant L sub 0 and the integral Lipschitz constant L sub 1. The stability bound is further dependent on the scalar factor delta, which measures the non commutativity between the operators S and T

4. This is good news. It means that, under mild conditions, algebraic filters can be made stable to perturbations

5. On the other hand, the upper bound is directly proportional to the variability of the filter as measured by the Lipschitz constant L_0 and,

6. more importantly, to the integral Lipschitz constant L_1, which controls the length of the interval in which the filter can vary. Alas, we encounter the same stability-discriminability tradeoff we have observed for graph and graphon filters

7. As for the commutativity factor, it affects the stability constant, but does not generate instability.

Slide 6: Stability of Algebraic Neural Networks, Single Layer

1. Next, we analyze the stability of an algebraic neural network layer.

2. For a formal statement, let Phi_l of (S,X) and Phi_l(S_tilde,X) be the operators associated with a layer l of an algebraic neural network.

3. Then, if the filters p are Lipschitz and integral Lipschitz,

4. The difference between the outputs of the AlgNNs on S and S_tilde can be upper bounded.

5. Note that the stability bound is the exact same bound we obtained for algebraic filters. Which is good news, because it means that algebraic neural networks can be made stable to perturbations.

6. However, due to the same stability-discriminability tradeoff as the one we observe in algebraic filters, individual layers lose discriminability

7. But in the AlgNN, this is addressed by the nonlinearity, which mixes the frequency components and allows discriminability in subsequent layers

Slide 7: Stability of Algebraic Neural Networks, Multiple Layers

1. For AlgNNs to be stable and discriminative, we know that they must have multiple layers. But stacking layers change the stability bound slightly.

2. To see this, let Phi of S and x and Phi of S tilde and x now denote the operators associated with an algebraic neural network on L layers.

3. Then, if the filters p at each layer are Lipschitz and integral Lipschitz,

4. The difference between the outputs of the AlgNNs on S and S tilde can be upper bounded.

5. This stability bound is only a minor variation of the stability bound we derived for a single layer. It is the same stability bound, only scaled by the number of layers L.

Slide 8: Stability of Algebraic Neural Networks, Multiple Generators

1. The algebraic neural network stability result can be readily extended to the case in which there are M generators.

2. Formally, let Phi of S and x and Phi of S tilde and x now denote the operators associated with an algebraic neural network with M generators on L layers.

3. Then, if the filters p at each layer are Lipschitz and integral Lipschitz,

4. The difference between the outputs of the ANNs on S and S tilde can be upper bounded.

5. This stability bound is only a minor variation of the stability bounds we derived for a single ANN layer and for the multilayer ANN. It is the same stability bound, only scaled by the number of layers L and the number of generators M.

12.9 Spectral Representations

Slide 1: Spectral Representations-Title page

19. Central to the analysis of algebraic signal models is the notion of spectral or Fourier decompositions. In this lecture we introduce this notion based on the concept of decompositions in terms of irreducible subrepresentations

Slide 2: Recalling basic concepts

20. We start recalling that an algebraic model is a triple A coma M coma rho

21. Where A is an algebra, M is a vector space and rho is a homomorphism

22. And where the pair M coma rho is a representation of the algebra A

23. We highlight in particular that each representation of an algebra defines an algebraic signal model

24. We point out that given two representations (M_1 , \rho_1) and (M_2 , \rho_2) is possible to define a direct sum as (M_1 \oplus M_2 , \rho). This is the vector space in the sum representation is the direct sum of M_1 and M_2 and there is a homomorphism rho

25. The action of \rho(a) on (X_1 \oplus X_2) equals the direct sum of the action of \rho_1(a) on X_1 and \rho_2(a) on X_2

Slide 3: Subrepresentations and Irreducible representations

26. Before defining irreducibility we require to define formally subrepresentations

27. Let M coma rho be a representation of the algebra A

28. Then, a representation U coma rho of the same algebra A is a subrepresentation of M coma rho

29. If the space U is contained in the space M and U is invariant under the action of rho, this is any filtered element of U is again in U

30. Now we can introduce the notion of irreducible representation

31. A representation M coma rho, where M is different from the zero vector space, is irreducible or simple

32. If the only sub-representations of M coma rho are zero coma rho or M coma rho itself

Slide 4: Intertwining operators

33. Now we introduce the notion of homorphism or intertwining operators between representations, which will allow us to state comparisons between them

34. Let (M_1 , rho_1) and (M_2 , \rho_2) be two representations of the same algebra A

35. A homomorphism or intertwining operator phi acting from M one to M two is a linear operator that commutes with the action of the algebra

36. This is phi of rho one of a on v equals rho two of a on phi of v

37. Additionally we will refer to phi as an isomorphism of representations if phi is in addition an isomorphism of vector spaces

38. We indicate that two representations are isomorphic using the congruence symbol

Slide 5: Fourier Decompositions

39. Now we introduce the main concept of this lecture, that one of Fourier decomposition

40. For an algebraic signal model (A,M,rho) we say that there is a spectral or Fourier decomposition of the representation (M,rho) if

41. (M,rho) is isomorphic to a direct sum of irreducible subrepresentations (U_i, phi_i)

42. Where m(U_i,M) indicates the number of irreducible subrepresentations of (M,rho) that are isomorphic to (U_i,phi_i)

43. And any signal X in M can be represented by the function \Delta that maps X into the direct sum of the spaces U_i as X-hat

44. The projection of X-hat in each space U_i are the Fourier components of X and are represented by X-hat of i

Slide 6: Fourier Decompositions highlights

45. Notice that the Fourier decomposition is defined by two sums

46. Indicated in blue and green color respectively

47. The first sum, in blue, is performed on the non isomorphic subrepresentations

48. While the sum in green is performed over all the subrepresentations that are isomorphic

49. Notice also that the external sum, in blue, indicates the frequencies while the internal sum, in green, the components associated to a given frequency

50. The operator Delta, who maps any signal x into its Fourier decomposition is an intertwining operator, which means it interchanges with the action of any filter a in the algebra. This also allows one to write the action of a filter on a signal using the action of filter on the signal in the spectral domain

Slide 7: Fourier Decompositions highlights

51. The projection of a filtered signal on each space U_i can be written in terms of the action of a homomorphism phi_i on x-hat of i.

52. And the collection of the projections of the filtered signal rho(a)x on U_i for all i defines the spectral representation of the filter

53. It is worth pointing out that the action of phi_i on x-hat of i is performed by means of different operations depending on the dimension of the spaces U_i

54. In particular we can see that if the dimension of U_i equals one, the operator phi_i of a is a scalar while if the dimension of U_i is greater than one and finite phi_i of a is a matrix

Slide 8: Remarks

55. Now we make some remarks. First we highlight that the link between the algebra and any Fourier decomposition is exclusively given by the homomorphism phi_i

56. And as a consequence it is not possible by selecting filters in the algebra to modify the spaces U_i in a Fourier decomposition of a given representation

57. Then, there are two sources of differences between the operators rho of a and rho tilda of a associated to the representations M coma rho and M coma rho tilda

58. One source of difference that can be modified by selecting subset of the algebra and this is something that is embedded in the homomorphisms phi_i and phi tilda sub i

59. While there is a second source of difference associated with the spaces U_i and U tilda sub i that cannot be modified, by any selection of the algebraic filters in the algebra.

Slide 9: Examples

60. Finally, we mention a couple of examples where we put more in concrete the notion of spectral representations. We start with traditional CNNs, where filtering is defined by the algebra of polynomials modulo t^N-1

61. The spectral representation of the filtering of a signal X is expressed in terms of the maps phi_i, associated to the irreducible subrepresentations, and which in this case

62. Are the eigenvalues associated to the column vectors u_i of the DFT matrix

63. For GNNs we consider the polynomial algebra with single variable t

64. Then, the spectral representation of the filtering of a signal X is expressed again in terms of the maps phi_i which in this case

65. Are the eigenvalues of the graph matrix S while the unit vectors u_i are the eigenvectors of the operator S