What is Tame Hypergraph Regularity About?

Henry Towsner

Introduction

This presentation looks best in full-screen mode.

This presentation looks best in full-screen mode.

If you aren't sure how to do that, you can probably find a command named something like "Full Screen" or "Presentation" in the "View" menu.

Tame hypergraph regularity is a recent area exploring the intersection of some ideas from graph theory with model theory.

This is a new, fairly technical area, but you don't need all the details to understand the main ideas.

Tame hypergraph regularity is a recent area exploring the intersection of some ideas from graph theory with model theory.

This is a new, fairly technical area, but you don't need all the details to understand the main ideas.

The goal of these slides is to explain the motivation and current state of knowledge in the area (in early 2023).

Tame hypergraph regularity is a recent area exploring the intersection of some ideas from graph theory with model theory.

This is a new, fairly technical area, but you don't need all the details to understand the main ideas.

The goal of these slides is to explain the motivation and current state of knowledge in the area (in early 2023).

"Tame hypergraph regularity" was inspired by the discovery of "tame graph regularity", so I should explain that first.

Tame Graph Regularity: The idea

I'm going to describe a gambling game. We'll start with two large finite sets, X and Y—they might both be "the set of natural numbers less than a million", for instance.

I'm going to describe a gambling game. We'll start with two large finite sets, X and Y—they might both be "the set of natural numbers less than a million", for instance. (Usually X and Y will be the same set, but it's convenient to include the option for them to be different.)

I'm going to describe a gambling game. We'll start with two large finite sets, X and Y—they might both be "the set of natural numbers less than a million", for instance. (Usually X and Y will be the same set, but it's convenient to include the option for them to be different.)

Later the house will pick two numbers, x from X and y from Y, and we need to decide which pairs are "good".

I'm going to describe a gambling game. We'll start with two large finite sets, X and Y—they might both be "the set of natural numbers less than a million", for instance. (Usually X and Y will be the same set, but it's convenient to include the option for them to be different.)

Later the house will pick two numbers, x from X and y from Y, and we need to decide which pairs are "good". Some representative examples we might choose are:

- A pair is good if their sum is even.

I'm going to describe a gambling game. We'll start with two large finite sets, X and Y—they might both be "the set of natural numbers less than a million", for instance. (Usually X and Y will be the same set, but it's convenient to include the option for them to be different.)

Later the house will pick two numbers, x from X and y from Y, and we need to decide which pairs are "good". Some representative examples we might choose are:

- A pair is good if their sum is even.
- A pair is good if x is smaller than y.

I'm going to describe a gambling game. We'll start with two large finite sets, X and Y—they might both be "the set of natural numbers less than a million", for instance. (Usually X and Y will be the same set, but it's convenient to include the option for them to be different.)

Later the house will pick two numbers, x from X and y from Y, and we need to decide which pairs are "good". Some representative examples we might choose are:

- A pair is good if their sum is even.
- A pair is good if x is smaller than y.
- In advance, we flip a coin for each pair, and make a big table of the results as a reference. A pair is good if the coin we flipped for that pair came up heads.

Once we've picked our sets X and Y and our good pairs, we're ready to play.

Once we've picked our sets X and Y and our good pairs, we're ready to play. First, the house picks random numbers x from X and y from Y and keeps them secret.

Once we've picked our sets X and Y and our good pairs, we're ready to play. First, the house picks random numbers x from X and y from Y and keeps them secret. Then we get to ask a small number of questions about x and y.

Once we've picked our sets X and Y and our good pairs, we're ready to play. First, the house picks random numbers x from X and y from Y and keeps them secret. Then we get to ask a small number of questions about x and y. And finally, we guess whether whether x and y are a good pair. We're hoping to know the answer, or at least—since this is ostensibly a gambling game—get it right most of the time.

Once we've picked our sets X and Y and our good pairs, we're ready to play. First, the house picks random numbers x from X and y from Y and keeps them secret. Then we get to ask a small number of questions about x and y. And finally, we guess whether whether x and y are a good pair. We're hoping to know the answer, or at least—since this is ostensibly a gambling game—get it right most of the time.

It's important that the the number of questions is much smaller than the sizes of X and Y, since if we had enough to pin down x and y exactly, the game wouldn't be very interesting.

Here's the crucial rule: the way we ask questions is that you ask questions about x, and meanwhile I ask questions about y, and then once we're both done asking questions, we get to compare the answers we got and decide whether we think it's a good pair.

Here's the crucial rule: the way we ask questions is that you ask questions about x, and meanwhile I ask questions about y, and then once we're both done asking questions, we get to compare the answers we got and decide whether we think it's a good pair. Neither of us can ask questions about both at once—if we could, we'd just ask if they're a good pair.

Here's the crucial rule: the way we ask questions is that you ask questions about x, and meanwhile I ask questions about y, and then once we're both done asking questions, we get to compare the answers we got and decide whether we think it's a good pair. Neither of us can ask questions about both at once—if we could, we'd just ask if they're a good pair. And I can't decide what questions to ask about y based on the answers to your questions about x, or vice-versa.

Here's the crucial rule: the way we ask questions is that you ask questions about x, and meanwhile I ask questions about y, and then once we're both done asking questions, we get to compare the answers we got and decide whether we think it's a good pair. Neither of us can ask questions about both at once—if we could, we'd just ask if they're a good pair. And I can't decide what questions to ask about y based on the answers to your questions about x, or vice-versa. We do get to confer in advance about our strategy, though.

Here's the crucial rule: the way we ask questions is that you ask questions about x, and meanwhile I ask questions about y, and then once we're both done asking questions, we get to compare the answers we got and decide whether we think it's a good pair. Neither of us can ask questions about both at once—if we could, we'd just ask if they're a good pair. And I can't decide what questions to ask about y based on the answers to your questions about x, or vice-versa. We do get to confer in advance about our strategy, though.

The question we're interested in is:

> *Which choices of the good pairs let us do well at this game? That is, when can we, with a small number of questions, figure out if a pair is good most of the time?*

Let's think through how this works in some examples.

Let's think through how this works in some examples.

Suppose that x, y is a good pair exactly when $x + y$ is even.

Let's think through how this works in some examples.

Suppose that x, y is a good pair exactly when $x + y$ is even. Then we have a great strategy: you ask if x is even and I ask if y is even.

Let's think through how this works in some examples.

Suppose that x, y is a good pair exactly when $x + y$ is even. Then we have a great strategy: you ask if x is even and I ask if y is even.

The answers to those questions are enough for us to figure out if x, y is a good pair. If x and y are both even or both odd, it's a good pair. If one is even and one is not, it's not a good pair.

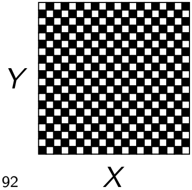
Let's think through how this works in some examples.

Suppose that x, y is a good pair exactly when $x + y$ is even. Then we have a great strategy: you ask if x is even and I ask if y is even.

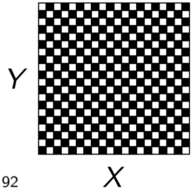
The answers to those questions are enough for us to figure out if x, y is a good pair. If x and y are both even or both odd, it's a good pair. If one is even and one is not, it's not a good pair.

So in this case, we each ask one question and then together we know the answer with complete certainty.

Here's a graphical representation of what's going on.

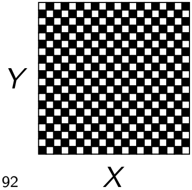


Here's a graphical representation of what's going on. The sides represent the sets X and Y and the darkened boxes are the good pairs.



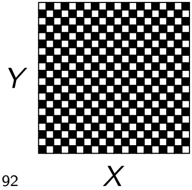
Here's a graphical representation of what's going on. The sides represent the sets X and Y and the darkened boxes are the good pairs.

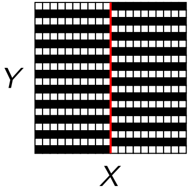
You ask whether x is even.



Here's a graphical representation of what's going on. The sides represent the sets X and Y and the darkened boxes are the good pairs.

You ask whether x is even. We'll reorganize the X axis based on the answer: we draw a red line through the middle and place the even columns on the left, and the odd columns on the right.

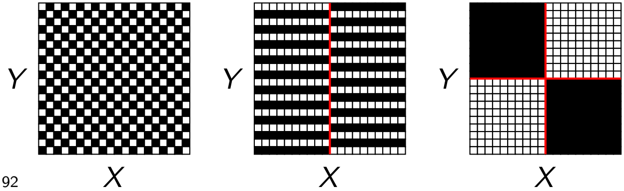




Here's a graphical representation of what's going on. The sides represent the sets X and Y and the darkened boxes are the good pairs.

You ask whether x is even. We'll reorganize the X axis based on the answer: we draw a red line through the middle and place the even columns on the left, and the odd columns on the right.

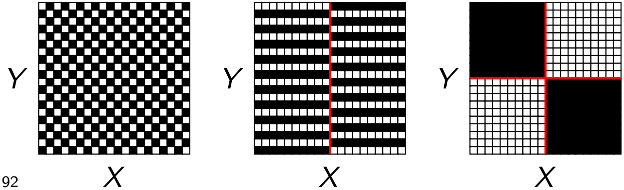
Meanwhile, I ask whether y is even, and again we reorganize the grid based on the answer: we put the even y's on top and the odd ones on the bottom and separate them with a red line, because we know which side of the red line we're on.



Here's a graphical representation of what's going on. The sides represent the sets X and Y and the darkened boxes are the good pairs.

You ask whether x is even. We'll reorganize the X axis based on the answer: we draw a red line through the middle and place the even columns on the left, and the odd columns on the right.

Meanwhile, I ask whether y is even, and again we reorganize the grid based on the answer: we put the even y's on top and the odd ones on the bottom and separate them with a red line, because we know which side of the red line we're on. The fact that we can win this game is reflected in the fact that (after rearranging the rows and columns to reflect our questions), the red lines divide the possible pairs into rectangles each of which is either all black (all good pairs) or all white (no good pairs).



How about if the rule is that x, y is a good pair exactly when $x < y$?

How about if the rule is that x, y is a good pair exactly when $x < y$?

We still have a strategy, though it's not quite as good.

How about if the rule is that x, y is a good pair exactly when $x < y$?

We still have a strategy, though it's not quite as good. Say we each get to ask two questions. First you ask if x is in the top half or the bottom half. Then you ask if it's in the top quarter or bottom quarter of whichever half it's in.

How about if the rule is that x, y is a good pair exactly when $x < y$?

We still have a strategy, though it's not quite as good. Say we each get to ask two questions. First you ask if x is in the top half or the bottom half. Then you ask if it's in the top quarter or bottom quarter of whichever half it's in. I ask the same questions about y.

How about if the rule is that x, y is a good pair exactly when $x < y$?

We still have a strategy, though it's not quite as good. Say we each get to ask two questions. First you ask if x is in the top half or the bottom half. Then you ask if it's in the top quarter or bottom quarter of whichever half it's in. I ask the same questions about y.

If we learn, say, that x is in the bottom quarter while y is in the second smallest quarter, we know that $x < y$, so it's a good pair.

How about if the rule is that x, y is a good pair exactly when $x < y$?

We still have a strategy, though it's not quite as good. Say we each get to ask two questions. First you ask if x is in the top half or the bottom half. Then you ask if it's in the top quarter or bottom quarter of whichever half it's in. I ask the same questions about y.

If we learn, say, that x is in the bottom quarter while y is in the second smallest quarter, we know that $x < y$, so it's a good pair. Similarly, if we learn that y is in the bottom half and x is in the top half, we know that $y < x$, so it's not a good pair.

How about if the rule is that x, y is a good pair exactly when $x < y$?

We still have a strategy, though it's not quite as good. Say we each get to ask two questions. First you ask if x is in the top half or the bottom half. Then you ask if it's in the top quarter or bottom quarter of whichever half it's in. I ask the same questions about y.

If we learn, say, that x is in the bottom quarter while y is in the second smallest quarter, we know that $x < y$, so it's a good pair. Similarly, if we learn that y is in the bottom half and x is in the top half, we know that $y < x$, so it's not a good pair.

If we're unlucky, though, x and y belong to the same quarter, and then we're not sure if the pair is good.

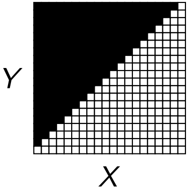
How about if the rule is that x, y is a good pair exactly when $x < y$?

We still have a strategy, though it's not quite as good. Say we each get to ask two questions. First you ask if x is in the top half or the bottom half. Then you ask if it's in the top quarter or bottom quarter of whichever half it's in. I ask the same questions about y.

If we learn, say, that x is in the bottom quarter while y is in the second smallest quarter, we know that $x < y$, so it's a good pair. Similarly, if we learn that y is in the bottom half and x is in the top half, we know that $y < x$, so it's not a good pair.

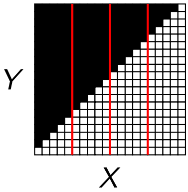
If we're unlucky, though, x and y belong to the same quarter, and then we're not sure if the pair is good. But that only happens a quarter of the time. So in this case, *most* of the time we know the answer after asking a few questions.

Here's the picture for $x < y$.



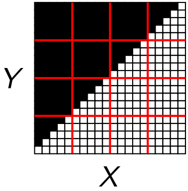
Here's the picture for $x < y$.

The answers to your two questions divide the X axis into four quarters. (As it happens, the columns are already in the right positions this time.)



Here's the picture for $x < y$.

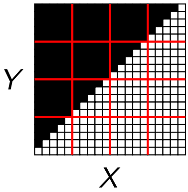
The answers to your two questions divide the X axis into four quarters.
(As it happens, the columns are already in the right positions this time.)
The answers to the my two questions divide the Y axis into four quarters.



Here's the picture for $x < y$.

The answers to your two questions divide the X axis into four quarters. (As it happens, the columns are already in the right positions this time.) The answers to the my two questions divide the Y axis into four quarters.

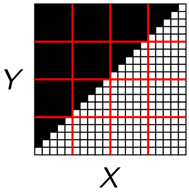
The red lines, which separate the axes based on the information you have after asking your questions, divide *most* of the picture into rectangles which are either all black or all white.



Here's the picture for $x < y$.

The answers to your two questions divide the X axis into four quarters. (As it happens, the columns are already in the right positions this time.) The answers to the my two questions divide the Y axis into four quarters.

The red lines, which separate the axes based on the information you have after asking your questions, divide *most* of the picture into rectangles which are either all black or all white. But there are still the areas along the diagonal which are mixed.



What if the good pairs were themselves chosen randomly: we just have a big table of good pairs generated by flipping coins?

What if the good pairs were themselves chosen randomly: we just have a big table of good pairs generated by flipping coins?

Then there's not much we can do:

What if the good pairs were themselves chosen randomly: we just have a big table of good pairs generated by flipping coins?

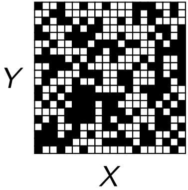
Then there's not much we can do: probably (as long as the coin flips that generated the table didn't do something very unlikely) asking a few questions won't help.

What if the good pairs were themselves chosen randomly: we just have a big table of good pairs generated by flipping coins?

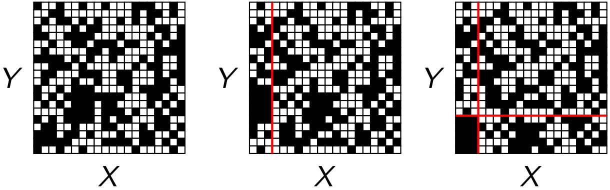
Then there's not much we can do: probably (as long as the coin flips that generated the table didn't do something very unlikely) asking a few questions won't help.

About half the pairs are good, and after asking a bunch of questions, we'll still probably think there's about a fifty percent chance that we're dealing with a good pair.

The random table might look something like this.

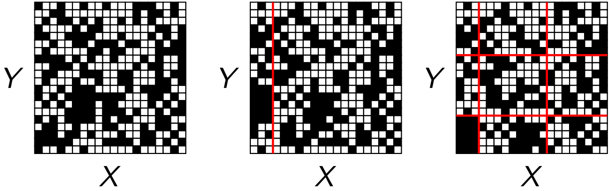


The random table might look something like this. We could ask questions that try to pin down rectangular clusters, but those clusters are all small: one answer to a question helps a lot, but most of the time, we'll get the unhelpful answer.



The random table might look something like this. We could ask questions that try to pin down rectangular clusters, but those clusters are all small: one answer to a question helps a lot, but most of the time, we'll get the unhelpful answer.

But if we ask questions that meaningfully divide up the region, we're left with recangles that are still a jumbled mix of good and bad pairs.



The discovery that leads to the idea of “tame graph regularity” is that these three examples represent three different “paradigms” for how this game can play out.

The discovery that leads to the idea of "tame graph regularity" is that these three examples represent three different "paradigms" for how this game can play out.

The two theorems that got the area started show these paradigms correspond to *model theoretic dividing lines*.

The discovery that leads to the idea of “tame graph regularity” is that these three examples represent three different “paradigms” for how this game can play out.

The two theorems that got the area started show these paradigms correspond to *model theoretic dividing lines*. Dividing lines are properties that separate structures which are, in some way, “simple”, from ones which are somehow “complicated”.

The discovery that leads to the idea of "tame graph regularity" is that these three examples represent three different "paradigms" for how this game can play out.

The two theorems that got the area started show these paradigms correspond to *model theoretic dividing lines*. Dividing lines are properties that separate structures which are, in some way, "simple", from ones which are somehow "complicated".

The dividing lines here were already well known when these theorems were proven—indeed, arguably the two most important ones. So what we learned from the discovery of tame graph regularity is that this game gives us an new way of looking at these existing dividing lines.

The discovery that leads to the idea of "tame graph regularity" is that these three examples represent three different "paradigms" for how this game can play out.

The two theorems that got the area started show these paradigms correspond to *model theoretic dividing lines*. Dividing lines are properties that separate structures which are, in some way, "simple", from ones which are somehow "complicated".

The dividing lines here were already well known when these theorems were proven—indeed, arguably the two most important ones. So what we learned from the discovery of tame graph regularity is that this game gives us an new way of looking at these existing dividing lines.

Let's talk about precisely how we define these "paradigms" and what the theorems characterizing them say.

Tame Graph Regularity: Approximable rules

The simpler dividing line to explain is the one separating the "$x < y$" from the example where we generated our table randomly.

The simpler dividing line to explain is the one separating the "$x < y$" from the example where we generated our table randomly.

First, we should introduce some of the standard terminology, and pin down our setting a little more carefully.

The simpler dividing line to explain is the one separating the "$x < y$" from the example where we generated our table randomly.

First, we should introduce some of the standard terminology, and pin down our setting a little more carefully.

We've been talking vaguely about having sets X and Y and a set of good pairs. The technical term for this is a *graph*.

The simpler dividing line to explain is the one separating the "$x < y$" from the example where we generated our table randomly.

First, we should introduce some of the standard terminology, and pin down our setting a little more carefully.

We've been talking vaguely about having sets X and Y and a set of good pairs. The technical term for this is a *graph*. More precisely, this is a *bipartite graph*—the distinct sets X and Y are the "two parts". But since these are the only kinds of graphs we'll talk about, I'll just call them graphs. We'll assume the sets X and Y are finite.

The simpler dividing line to explain is the one separating the "$x < y$" from the example where we generated our table randomly.

First, we should introduce some of the standard terminology, and pin down our setting a little more carefully.

We've been talking vaguely about having sets X and Y and a set of good pairs. The technical term for this is a *graph*. More precisely, this is a *bipartite graph*—the distinct sets X and Y are the "two parts". But since these are the only kinds of graphs we'll talk about, I'll just call them graphs. We'll assume the sets X and Y are finite.

Instead of talking about "good pairs", we'll use the standard terminology and call them *edges*.

The simpler dividing line to explain is the one separating the "$x < y$" from the example where we generated our table randomly.

First, we should introduce some of the standard terminology, and pin down our setting a little more carefully.

We've been talking vaguely about having sets X and Y and a set of good pairs. The technical term for this is a *graph*. More precisely, this is a *bipartite graph*—the distinct sets X and Y are the "two parts". But since these are the only kinds of graphs we'll talk about, I'll just call them graphs. We'll assume the sets X and Y are finite.

Instead of talking about "good pairs", we'll use the standard terminology and call them *edges*. So formally, a graph is three things, (X, Y, E) where X and Y are sets and $E \subseteq X \times Y$ is the set of edges.

You might have noticed that our examples weren't exactly about individual graphs—they were more like families of similar graphs.

You might have noticed that our examples weren't exactly about individual graphs—they were more like families of similar graphs. Instead of thinking about the specific graph where X and Y are the numbers up to a million and the edges are when $x < y$, we could say "for any n, we have a graph where X and Y are the numbers up to n and the edges are when $x < y$".

You might have noticed that our examples weren't exactly about individual graphs—they were more like families of similar graphs. Instead of thinking about the specific graph where X and Y are the numbers up to a million and the edges are when $x < y$, we could say "for any n, we have a graph where X and Y are the numbers up to n and the edges are when $x < y$".

Definition

A *rule* is a set of graphs.

This isn't standard terminology, but it's convenient for us. (The standard term is a *hereditary class of graphs*.)

You might have noticed that our examples weren't exactly about individual graphs—they were more like families of similar graphs. Instead of thinking about the specific graph where X and Y are the numbers up to a million and the edges are when $x < y$, we could say "for any n, we have a graph where X and Y are the numbers up to n and the edges are when $x < y$".

Definition

A *rule* is a set of graphs.

This isn't standard terminology, but it's convenient for us. (The standard term is a *hereditary class of graphs*.)

Some examples of rules are:

- $\mathcal{R}_{\mathrm{even}}$: X and Y are the set of numbers up to n for some n, and the edges are where $x + y$ is even,

You might have noticed that our examples weren't exactly about individual graphs—they were more like families of similar graphs. Instead of thinking about the specific graph where X and Y are the numbers up to a million and the edges are when $x < y$, we could say "for any n, we have a graph where X and Y are the numbers up to n and the edges are when $x < y$".

Definition

A *rule* is a set of graphs.

This isn't standard terminology, but it's convenient for us. (The standard term is a *hereditary class of graphs*.)

Some examples of rules are:

- $\mathcal{R}_{\mathrm{even}}$: X and Y are the set of numbers up to n for some n, and the edges are where $x + y$ is even,
- $\mathcal{R}_{<}$: X and Y are the set of numbers up to n for some n, and the edges are where $x < y$,

You might have noticed that our examples weren't exactly about individual graphs—they were more like families of similar graphs. Instead of thinking about the specific graph where X and Y are the numbers up to a million and the edges are when $x < y$, we could say "for any n, we have a graph where X and Y are the numbers up to n and the edges are when $x < y$".

Definition

A *rule* is a set of graphs.

This isn't standard terminology, but it's convenient for us. (The standard term is a *hereditary class of graphs*.)

Some examples of rules are:

- $\mathcal{R}_{\mathrm{even}}$: X and Y are the set of numbers up to n for some n, and the edges are where $x + y$ is even,
- $\mathcal{R}_{<}$: X and Y are the set of numbers up to n for some n, and the edges are where $x < y$,
- $\mathcal{R}_{\mathrm{rand}}$: For each n, we will generate a random graph where X and Y are both the set of numbers up to n, and we determine which pairs are edges by flipping a separate coin for each pair.

Here's the property that will distinguish examples like $x < y$ from the randomly generated rule:

Definition

A rule $\mathcal{R}$ is *approximable* if for every $\epsilon > 0$, there is a number N so that for any graph (X, Y, E) in $\mathcal{R}$, if $x \in X$ and $y \in Y$ are chosen randomly and we ask N questions about x and y separately, the probability that we correctly guess whether the pair is an edge is at least $1 - \epsilon$.

Here's the property that will distinguish examples like $x < y$ from the randomly generated rule:

Definition

A rule $\mathcal{R}$ is *approximable* if for every $\epsilon > 0$, there is a number N so that for any graph (X, Y, E) in $\mathcal{R}$, if $x \in X$ and $y \in Y$ are chosen randomly and we ask N questions about x and y separately, the probability that we correctly guess whether the pair is an edge is at least $1 - \epsilon$.

If we think of this as a gambling game, we pay \$1 to play the game and the house has set a payout of $\$1+\epsilon$ if we get the answer right.

Here's the property that will distinguish examples like $x < y$ from the randomly generated rule:

Definition

A rule $\mathcal{R}$ is *approximable* if for every $\epsilon > 0$, there is a number N so that for any graph (X, Y, E) in $\mathcal{R}$, if $x \in X$ and $y \in Y$ are chosen randomly and we ask N questions about x and y separately, the probability that we correctly guess whether the pair is an edge is at least $1 - \epsilon$.

If we think of this as a gambling game, we pay \$1 to play the game and the house has set a payout of \$1$+\epsilon$ if we get the answer right. Then we decide how many questions we need to ask to achieve that accuracy.

Here's the property that will distinguish examples like $x < y$ from the randomly generated rule:

Definition

A rule $\mathcal{R}$ is *approximable* if for every $\epsilon > 0$, there is a number N so that for any graph (X, Y, E) in $\mathcal{R}$, if $x \in X$ and $y \in Y$ are chosen randomly and we ask N questions about x and y separately, the probability that we correctly guess whether the pair is an edge is at least $1 - \epsilon$.

If we think of this as a gambling game, we pay \$1 to play the game and the house has set a payout of \$1$+\epsilon$ if we get the answer right. Then we decide how many questions we need to ask to achieve that accuracy. Then the house chooses a graph from $\mathcal{R}$ to play against (and tells us what the graph is)—in particular, it could be one where X and Y are much, much, much bigger than the number of questions we get.

Here's the property that will distinguish examples like $x < y$ from the randomly generated rule:

Definition

A rule $\mathcal{R}$ is *approximable* if for every $\epsilon > 0$, there is a number N so that for any graph (X, Y, E) in $\mathcal{R}$, if $x \in X$ and $y \in Y$ are chosen randomly and we ask N questions about x and y separately, the probability that we correctly guess whether the pair is an edge is at least $1 - \epsilon$.

If we think of this as a gambling game, we pay \$1 to play the game and the house has set a payout of \$1$+\epsilon$ if we get the answer right. Then we decide how many questions we need to ask to achieve that accuracy. Then the house chooses a graph from $\mathcal{R}$ to play against (and tells us what the graph is)—in particular, it could be one where X and Y are much, much, much bigger than the number of questions we get.

$\mathcal{R}$ is approximable exactly when this game is worth playing anyway: no matter how stingy the payout is, and no matter how hard the house tries to pick a rule from $\mathcal{R}$ that makes our life difficult, we can pick a number N which is big enough so that we'll make money on average.

The name “approximable” isn’t standard. The notion has appeared in various papers under various names, most of which are kind of ad hoc, and no widely accepted name has emerged, so “approximable” will do for us.

$\mathcal{R}_<$—the set of graphs where X and Y are sets of numbers less than n for some n and there's an edge when $x < y$—is approximable.

$\mathcal{R}_{<}$—the set of graphs where X and Y are sets of numbers less than n for some n and there's an edge when $x < y$—is approximable.

Say ϵ is 0.1, so we need to win more than 90% of the time to make a profit.

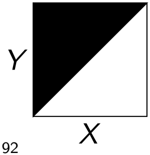
$\mathcal{R}_<$—the set of graphs where X and Y are sets of numbers less than n for some n and there's an edge when $x < y$—is approximable.

Say ϵ is 0.1, so we need to win more than 90% of the time to make a profit. Three questions each is enough.

$\mathcal{R}_<$—the set of graphs where X and Y are sets of numbers less than n for some n and there's an edge when $x < y$—is approximable.

Say ϵ is 0.1, so we need to win more than 90% of the time to make a profit. Three questions each is enough.

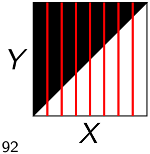
The house picks some very big n to be the size of X and Y.



$\mathcal{R}_{<}$—the set of graphs where X and Y are sets of numbers less than n for some n and there's an edge when $x < y$—is approximable.

Say ϵ is 0.1, so we need to win more than 90% of the time to make a profit. Three questions each is enough.

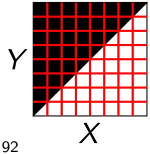
The house picks some very big n to be the size of X and Y. With three questions about X, we know which eighth of X the randomly chosen x belongs to.



$\mathcal{R}_<$—the set of graphs where X and Y are sets of numbers less than n for some n and there's an edge when $x < y$—is approximable.

Say ϵ is 0.1, so we need to win more than 90% of the time to make a profit. Three questions each is enough.

The house picks some very big n to be the size of X and Y. With three questions about X, we know which eighth of X the randomly chosen x belongs to. With three questions about Y, we know which eighth y belongs to.

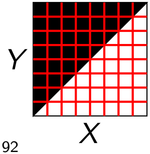


$\mathcal{R}_<$—the set of graphs where X and Y are sets of numbers less than n for some n and there's an edge when $x < y$—is approximable.

Say ϵ is 0.1, so we need to win more than 90% of the time to make a profit. Three questions each is enough.

The house picks some very big n to be the size of X and Y. With three questions about X, we know which eighth of X the randomly chosen x belongs to. With three questions about Y, we know which eighth y belongs to.

$7/8$ of the time, x and y belong to different octiles, and we know for sure if the pair is an edge. The remaining $1/8$ of the time, we have to risk a $50/50$ guess.

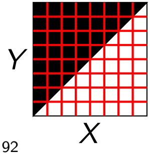


$\mathcal{R}_<$—the set of graphs where X and Y are sets of numbers less than n for some n and there's an edge when $x < y$—is approximable.

Say ϵ is 0.1, so we need to win more than 90% of the time to make a profit. Three questions each is enough.

The house picks some very big n to be the size of X and Y. With three questions about X, we know which eighth of X the randomly chosen x belongs to. With three questions about Y, we know which eighth y belongs to.

$7/8$ of the time, x and y belong to different octiles, and we know for sure if the pair is an edge. The remaining $1/8$ of the time, we have to risk a $50/50$ guess. But in total, we get the right answer $15/16$ of the time—about 93%, which is good enough to make money on average even if we only win \$1.10 for every correct guess.



On the other hand, $\mathcal{R}_{\mathrm{rand}}$, where each graph is created by flipping coins to make a random graph, is not approximable.

On the other hand, $\mathcal{R}_{\mathrm{rand}}$, where each graph is created by flipping coins to make a random graph, is not approximable.

Indeed, with probability 1, $\mathcal{R}_{\mathrm{rand}}$ has the property that we can't even get the right answer a tiny bit more than half the time, no matter how many questions we decide to ask.

On the other hand, $\mathcal{R}_{\mathrm{rand}}$, where each graph is created by flipping coins to make a random graph, is not approximable.

Indeed, with probability 1, $\mathcal{R}_{\mathrm{rand}}$ has the property that we can't even get the right answer a tiny bit more than half the time, no matter how many questions we decide to ask.

So approximability is a property that distinguishes rules "like $x < y$" from "like a random graph".

It turns out that approximability is characterized by "omitted subgraphs".

It turns out that approximability is characterized by "omitted subgraphs".

To explain what an omitted subgraph is, it's easiest to start with an example.

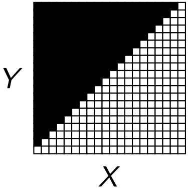
It turns out that approximability is characterized by "omitted subgraphs".

To explain what an omitted subgraph is, it's easiest to start with an example. The rule $x < y$ omits the subgraph .

It turns out that approximability is characterized by “omitted subgraphs”.

To explain what an omitted subgraph is, it’s easiest to start with an example. The rule $x < y$ omits the subgraph .

That is, if we pick two rows and two columns in



the subgrid we get never looks like (even if we reorder the rows and columns to try to make them match).

Of course, any finite graph omits subgraphs if they're big enough—say, bigger than the graph itself.

Of course, any finite graph omits subgraphs if they're big enough—say, bigger than the graph itself. So we care about when our graphs omit small subgraphs.

Of course, any finite graph omits subgraphs if they're big enough—say, bigger than the graph itself. So we care about when our graphs omit small subgraphs.

Definition

A rule $\mathcal{R}$ has *finite VC dimension* if there is some n so that every graph in $\mathcal{R}$ omits some $n \times n$ subgraph.

Of course, any finite graph omits subgraphs if they're big enough—say, bigger than the graph itself. So we care about when our graphs omit small subgraphs.

Definition

A rule $\mathcal{R}$ has *finite VC dimension* if there is some n so that every graph in $\mathcal{R}$ omits some $n \times n$ subgraph.

The name "VC dimension" is the standard one. (VC stands for "Vapnik–Chernovenkis", the people who first discovered it.)

Of course, any finite graph omits subgraphs if they're big enough—say, bigger than the graph itself. So we care about when our graphs omit small subgraphs.

Definition

A rule $\mathcal{R}$ has *finite VC dimension* if there is some n so that every graph in $\mathcal{R}$ omits some $n \times n$ subgraph.

The name "VC dimension" is the standard one. (VC stands for "Vapnik–Chernovenkis", the people who first discovered it.) It's a useful notion that's been rediscovered several times in several fields.

The first theorem that could be called a tame graph regularity result is:

Theorem (Lovász-Szegedy)

$\mathcal{R}$ has finite VC dimension exactly when $\mathcal{R}$ is approximable.

The first theorem that could be called a tame graph regularity result is:

Theorem (Lovász-Szegedy)

$\mathcal{R}$ has finite VC dimension exactly when $\mathcal{R}$ is approximable.

This is the prototype for what a tame graph (or hypergraph) regularity theorem should look like:

The first theorem that could be called a tame graph regularity result is:

Theorem (Lovász-Szegedy)

$\mathcal{R}$ has finite VC dimension exactly when $\mathcal{R}$ is approximable.

This is the prototype for what a tame graph (or hypergraph) regularity theorem should look like: we have an equivalence between a property about what kinds of subgraphs appear in $\mathcal{R}$ on the one hand and a property about how well you can win our game with graphs from $\mathcal{R}$ on the other.

The first theorem that could be called a tame graph regularity result is:

Theorem (Lovász-Szegedy)

$\mathcal{R}$ has finite VC dimension exactly when $\mathcal{R}$ is approximable.

This is the prototype for what a tame graph (or hypergraph) regularity theorem should look like: we have an equivalence between a property about what kinds of subgraphs appear in $\mathcal{R}$ on the one hand and a property about how well you can win our game with graphs from $\mathcal{R}$ on the other.

Finite VC dimension is such a well-studied notion, and many other equivalences are known, so approximability is one more way that finite VC dimension distinguishes simple sets of graphs from complicated ones.

A small confession: I'm glossing over a technicality here.

To actually get the equivalence, we have to clarify that the house is allowed to pick the probability distribution on X and Y (and tell us at the same time the house tells us which graph (X, Y, E) has been picked)

A small confession: I'm glossing over a technicality here.

To actually get the equivalence, we have to clarify that the house is allowed to pick the probability distribution on X and Y (and tell us at the same time the house tells us which graph (X, Y, E) has been picked)

I'll continue to ignore this detail below.

Tame Graph Regularity: Unary error

Next, let's see how $\mathcal{R}_{\mathrm{even}}$ is even nicer than $\mathcal{R}_{<}$.

Next, let's see how $\mathcal{R}_{\mathrm{even}}$ is even nicer than $\mathcal{R}_{<}$.

We could change the rules so that we're required to get the answer *exactly* right, not merely get it right most of the time. With $\mathcal{R}_{even}$, we'd still win under those rules.

Next, let's see how $\mathcal{R}_{\mathrm{even}}$ is even nicer than $\mathcal{R}_{<}$.

We could change the rules so that we're required to get the answer *exactly* right, not merely get it right most of the time. With $\mathcal{R}_{even}$, we'd still win under those rules.

But that turns out to me too much to ask: it's a very specialized property that happens only for very simple rules.

Next, let's see how $\mathcal{R}_{\mathrm{even}}$ is even nicer than $\mathcal{R}_{<}$.

We could change the rules so that we're required to get the answer *exactly* right, not merely get it right most of the time. With $\mathcal{R}_{even}$, we'd still win under those rules.

But that turns out to me too much to ask: it's a very specialized property that happens only for very simple rules.

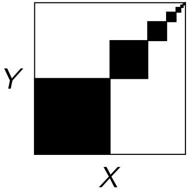
To really illustrate the interesting property that makes $\mathcal{R}_{\mathrm{even}}$ nicer than $\mathcal{R}_{<}$, it will be helpful to have a new example that's going to be nice in the same way as $\mathcal{R}_{\mathrm{even}}$, but better illustrates what sort of behavior is allowed.

In our new example, $\mathcal{R}_{\mathrm{digit}}$, X and Y will be the set of numbers up to n where n is some power of 2.

In our new example, $\mathcal{R}_{\mathrm{digit}}$, X and Y will be the set of numbers up to n where n is some power of 2. When we pick x and y, we'll write them in binary, padding the left side with 0's so they have the same number of digits. We'll have an edge between x and y if the first (i.e. leftmost, highest place value) 0 occurs in the same place for both of them.

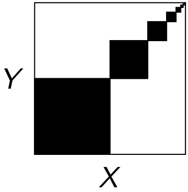
In our new example, $\mathcal{R}_{\mathrm{digit}}$, X and Y will be the set of numbers up to n where n is some power of 2. When we pick x and y, we'll write them in binary, padding the left side with 0's so they have the same number of digits. We'll have an edge between x and y if the first (i.e. leftmost, highest place value) 0 occurs in the same place for both of them.

We'll get an idea how this works much faster if we look at a picture.



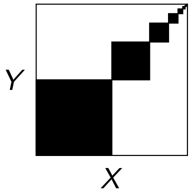
In our new example, $\mathcal{R}_{\mathrm{digit}}$, X and Y will be the set of numbers up to n where n is some power of 2. When we pick x and y, we'll write them in binary, padding the left side with 0's so they have the same number of digits. We'll have an edge between x and y if the first (i.e. leftmost, highest place value) 0 occurs in the same place for both of them.

We'll get an idea how this works much faster if we look at a picture. The big box of edges in the lower left corner is all the pairs where both x and y begin with a 0.



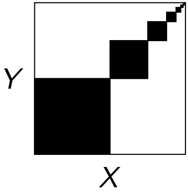
In our new example, $\mathcal{R}_{\mathrm{digit}}$, X and Y will be the set of numbers up to n where n is some power of 2. When we pick x and y, we'll write them in binary, padding the left side with 0's so they have the same number of digits. We'll have an edge between x and y if the first (i.e. leftmost, highest place value) 0 occurs in the same place for both of them.

We'll get an idea how this works much faster if we look at a picture. The big box of edges in the lower left corner is all the pairs where both x and y begin with a 0. The second, slightly smaller box of edges above it is the pairs where both x and y begin with 10.

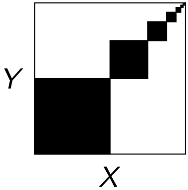


In our new example, $\mathcal{R}_{\text{digit}}$, X and Y will be the set of numbers up to n where n is some power of 2. When we pick x and y, we'll write them in binary, padding the left side with 0's so they have the same number of digits. We'll have an edge between x and y if the first (i.e. leftmost, highest place value) 0 occurs in the same place for both of them.

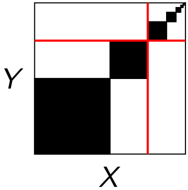
We'll get an idea how this works much faster if we look at a picture. The big box of edges in the lower left corner is all the pairs where both x and y begin with a 0. The second, slightly smaller box of edges above it is the pairs where both x and y begin with 10. And so on.



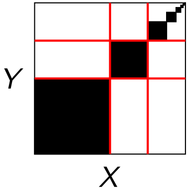
We can certainly approximate this. Suppose we each get two questions.



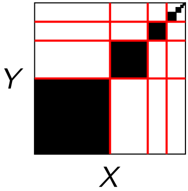
We can certainly approximate this. Suppose we each get two questions. You could first ask "is at least one of the first two digits of x a 0", and I ask the same about y.



We can certainly approximate this. Suppose we each get two questions. You could first ask "is at least one of the first two digits of x a 0", and I ask the same about y. When the answer was yes to one of these questions, we then ask whether the first digit was a 0.

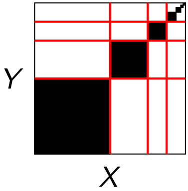


We can certainly approximate this. Suppose we each get two questions. You could first ask “is at least one of the first two digits of x a 0”, and I ask the same about y. When the answer was yes to one of these questions, we then ask whether the first digit was a 0. When the answer was no to the one of the initial questions, we instead ask whether the third digit was a 0.



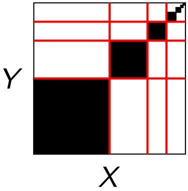
We can certainly approximate this. Suppose we each get two questions. You could first ask "is at least one of the first two digits of x a 0", and I ask the same about y. When the answer was yes to one of these questions, we then ask whether the first digit was a 0. When the answer was no to the one of the initial questions, we instead ask whether the third digit was a 0.

Most of the time, we know whether x, y is a good pair, and a small fraction of the time—when all four of our questions got answered "no"—we don't.

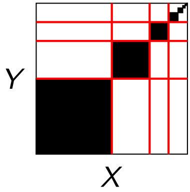


We can certainly approximate this. Suppose we each get two questions. You could first ask "is at least one of the first two digits of x a 0", and I ask the same about y. When the answer was yes to one of these questions, we then ask whether the first digit was a 0. When the answer was no to the one of the initial questions, we instead ask whether the third digit was a 0.

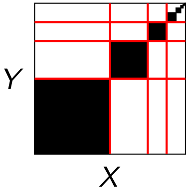
Most of the time, we know whether x, y is a good pair, and a small fraction of the time—when all four of our questions got answered "no"—we don't. Or we could notice that  is omitted and invoke the theorem characterizing approximability.



Like with $\mathcal{R}_<$, a small fraction of the time we'll get this wrong.

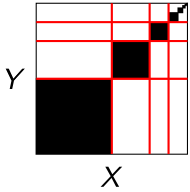


Like with $\mathcal{R}_<$, a small fraction of the time we'll get this wrong. But we can tell *before* we compare answers whether that's in danger of happening.



Like with $\mathcal{R}_<$, a small fraction of the time we'll get this wrong. But we can tell *before* we compare answers whether that's in danger of happening.

The only times we get the answer wrong are when we end up in the upper right corner. So if you give up every time your answers tell you we're in the far right segment, and I give up every time my answers tell me we're in the topmost segment, then we only give up a small fraction of the time, and whenever we haven't given up, we can be confident we have the right answer.



Our modified game works like this: like before, the house has set a small value ϵ, and we say what number N of questions we need to ask based on that. The house picks x and y randomly secretly, and you ask a small number of questions about x while I ask a small number of questions about y.

Our modified game works like this: like before, the house has set a small value ϵ, and we say what number N of questions we need to ask based on that. The house picks x and y randomly secretly, and you ask a small number of questions about x while I ask a small number of questions about y. The new rule is that, before we communicate, we each have the option of giving up.

Our modified game works like this: like before, the house has set a small value ϵ, and we say what number N of questions we need to ask based on that. The house picks x and y randomly secretly, and you ask a small number of questions about x while I ask a small number of questions about y. The new rule is that, before we communicate, we each have the option of giving up. We're each only allowed to give up ϵ of the time.

Our modified game works like this: like before, the house has set a small value ϵ, and we say what number N of questions we need to ask based on that. The house picks x and y randomly secretly, and you ask a small number of questions about x while I ask a small number of questions about y. The new rule is that, before we communicate, we each have the option of giving up. We're each only allowed to give up ϵ of the time. Any time neither of us gives up, we need to actually correctly guess whether the pair is an edge.

Our modified game works like this: like before, the house has set a small value ϵ, and we say what number N of questions we need to ask based on that. The house picks x and y randomly secretly, and you ask a small number of questions about x while I ask a small number of questions about y. The new rule is that, before we communicate, we each have the option of giving up. We're each only allowed to give up ϵ of the time. Any time neither of us gives up, we need to actually correctly guess whether the pair is an edge.

We say $\mathcal{R}$ *admits unary error* if we can win this harder game.

Our modified game works like this: like before, the house has set a small value ϵ, and we say what number N of questions we need to ask based on that. The house picks x and y randomly secretly, and you ask a small number of questions about x while I ask a small number of questions about y. The new rule is that, before we communicate, we each have the option of giving up. We're each only allowed to give up ϵ of the time. Any time neither of us gives up, we need to actually correctly guess whether the pair is an edge.

We say $\mathcal{R}$ *admits unary error* if we can win this harder game. The word "unary" here refers to the fact that we have to decide whether to give up individually, based on one person's information.

In our example, we *both* notice that we're in danger of being in the upper right corner.

In our example, we *both* notice that we're in danger of being in the upper right corner.

In general, though, the definition allows either of us to give up unilaterally (as long as we don't do it often).

In our example, we *both* notice that we're in danger of being in the upper right corner.

In general, though, the definition allows either of us to give up unilaterally (as long as we don't do it often).

Another confession: in the full definition, we don't actually have to get it exactly perfect even when we don't give up, we just have to be pretty confident we know what the answer is. This only comes up in more complicated examples, so we'll continue glossing over this detail.

We can identify which rules admit unary error by looking at omitted subgraphs.

We can identify which rules admit unary error by looking at omitted subgraphs.

To be approximable, a graph must omit *some* small subgraph. To admit unary error, it turns out it has to omit a *particular* small subgraph.

We can identify which rules admit unary error by looking at omitted subgraphs.

To be approximable, a graph must omit *some* small subgraph. To admit unary error, it turns out it has to omit a *particular* small subgraph.

Definition

The *half-graph of size k* is the $k \times k$ graph whose edges are where $x < y$. A graph is *k-stable* if it omits the half-graph of size k. $\mathcal{R}$ is stable if there is some k so that every graph in $\mathcal{R}$ is k-stable.

We can identify which rules admit unary error by looking at omitted subgraphs.

To be approximable, a graph must omit *some* small subgraph. To admit unary error, it turns out it has to omit a *particular* small subgraph.

Definition

The *half-graph of size k* is the $k \times k$ graph whose edges are where $x < y$. A graph is *k-stable* if it omits the half-graph of size k. $\mathcal{R}$ is stable if there is some k so that every graph in $\mathcal{R}$ is k-stable.

As we already noticed, $\mathcal{R}_{\text{digit}}$ is 2-stable: it omits the half-graph of size 2, .

One tame graph regularity theorem was a curiosity; two was a pattern.
The second theorem about tame graph regularity was:

Theorem (Malliaris–Shelah)

$\mathcal{R}$ is stable exactly when $\mathcal{R}$ admits unary error.

One tame graph regularity theorem was a curiosity; two was a pattern. The second theorem about tame graph regularity was:

Theorem (Malliaris–Shelah)

$\mathcal{R}$ is stable exactly when $\mathcal{R}$ admits unary error.

So $\mathcal{R}_<$ is not just any example of something that's approximable but doesn't admit unary error, it's in some sense the only example: anything which doesn't admit unary error will have something that looks like $\mathcal{R}_<$ inside it.

One tame graph regularity theorem was a curiosity; two was a pattern. The second theorem about tame graph regularity was:

Theorem (Malliaris–Shelah)

$\mathcal{R}$ is stable exactly when $\mathcal{R}$ admits unary error.

So $\mathcal{R}_<$ is not just any example of something that's approximable but doesn't admit unary error, it's in some sense the only example: anything which doesn't admit unary error will have something that looks like $\mathcal{R}_<$ inside it.

Stability is not merely *a* dividing line in model theory, it is the prototypical dividing line: much of modern model theory was originally developed by studying the ways stable graphs are well-behaved.

One tame graph regularity theorem was a curiosity; two was a pattern. The second theorem about tame graph regularity was:

Theorem (Malliaris–Shelah)

$\mathcal{R}$ is stable exactly when $\mathcal{R}$ admits unary error.

So $\mathcal{R}_<$ is not just any example of something that's approximable but doesn't admit unary error, it's in some sense the only example: anything which doesn't admit unary error will have something that looks like $\mathcal{R}_<$ inside it.

Stability is not merely *a* dividing line in model theory, it is the prototypical dividing line: much of modern model theory was originally developed by studying the ways stable graphs are well-behaved. Again, there were already many equivalent definitions of stability, so this was one more way to characterize a property we already know was important.

One tame graph regularity theorem was a curiosity; two was a pattern. The second theorem about tame graph regularity was:

Theorem (Malliaris–Shelah)

$\mathcal{R}$ is stable exactly when $\mathcal{R}$ admits unary error.

So $\mathcal{R}_<$ is not just any example of something that's approximable but doesn't admit unary error, it's in some sense the only example: anything which doesn't admit unary error will have something that looks like $\mathcal{R}_<$ inside it.

Stability is not merely *a* dividing line in model theory, it is the prototypical dividing line: much of modern model theory was originally developed by studying the ways stable graphs are well-behaved. Again, there were already many equivalent definitions of stability, so this was one more way to characterize a property we already know was important. (In this context, there's no short way to explain why the name "stable" makes sense, but there is another equivalent characterization which explains the name.)

An Aside: What is regularity?

Somehow we've been able to discuss the two results that motivate tame graph regularity without actually saying what regularity is.

Somehow we've been able to discuss the two results that motivate tame graph regularity without actually saying what regularity is.

"Regularity" is a reference to Szemerédi's Regularity Lemma.

Somehow we've been able to discuss the two results that motivate tame graph regularity without actually saying what regularity is.

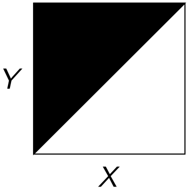
"Regularity" is a reference to Szemerédi's Regularity Lemma. Roughly speaking, in the context of the game we've been describing, it just says: "eventually you should stop asking questions and guess".

Szemerédi's Regularity Lemma is what we can say about any graph, not just approximable ones. What could we hope to do with a graph where we definitely can't win this gambling game?

Szemerédi's Regularity Lemma is what we can say about any graph, not just approximable ones. What could we hope to do with a graph where we definitely can't win this gambling game? Here's what we can hope to do: we'll ask our questions, and then at the end, we'll just say how likely we think it is that the pair is an edge.

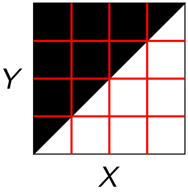
Szemerédi's Regularity Lemma is what we can say about any graph, not just approximable ones. What could we hope to do with a graph where we definitely can't win this gambling game? Here's what we can hope to do: we'll ask our questions, and then at the end, we'll just say how likely we think it is that the pair is an edge.

For instance, on a graph from $\mathcal{R}_<$, initially all we can say is that half the pairs are edges.



Szemerédi's Regularity Lemma is what we can say about any graph, not just approximable ones. What could we hope to do with a graph where we definitely can't win this gambling game? Here's what we can hope to do: we'll ask our questions, and then at the end, we'll just say how likely we think it is that the pair is an edge.

For instance, on a graph from $\mathcal{R}_<$, initially all we can say is that half the pairs are edges. After asking a few questions, we're probably in a situation where we can say either all or none of the pairs are edges, and there's a small chance the pair on the diagonal and we still only know that there's a fifty percent chance it's an edge.



On the other hand with a graph from $\mathcal{R}_{\mathrm{rand}}$, initially we think half the pairs are edges,

On the other hand with a graph from $\mathcal{R}_{\mathrm{rand}}$, initially we think half the pairs are edges, and after asking a bunch of questions, we still probably don't know anything more than that there's a fifty percent chance the pair is an edge.

On the other hand with a graph from $\mathcal{R}_{\mathrm{rand}}$, initially we think half the pairs are edges, and after asking a bunch of questions, we still probably don't know anything more than that there's a fifty percent chance the pair is an edge.

There's a term for what happened with $\mathcal{R}_{\mathrm{rand}}$: a graph is *quasirandom* if asking a few questions doesn't change the chance that we're looking at an edge. We started out thinking there was probability p that the pair is an edge, and after asking a few questions, we still (probably) think the chance is pretty close to p that the pair is an edge.

On the other hand with a graph from $\mathcal{R}_{\mathrm{rand}}$, initially we think half the pairs are edges, and after asking a bunch of questions, we still probably don't know anything more than that there's a fifty percent chance the pair is an edge.

There's a term for what happened with $\mathcal{R}_{\mathrm{rand}}$: a graph is *quasirandom* if asking a few questions doesn't change the chance that we're looking at an edge. We started out thinking there was probability p that the pair is an edge, and after asking a few questions, we still (probably) think the chance is pretty close to p that the pair is an edge.

When we generate a graph by flipping coins, the result is quasirandom.

On the other hand with a graph from $\mathcal{R}_{\mathrm{rand}}$, initially we think half the pairs are edges, and after asking a bunch of questions, we still probably don't know anything more than that there's a fifty percent chance the pair is an edge.

There's a term for what happened with $\mathcal{R}_{\mathrm{rand}}$: a graph is *quasirandom* if asking a few questions doesn't change the chance that we're looking at an edge. We started out thinking there was probability p that the pair is an edge, and after asking a few questions, we still (probably) think the chance is pretty close to p that the pair is an edge.

When we generate a graph by flipping coins, the result is quasirandom. Technically the graph with all edges and the graph with no edges are quasirandom, too: if you're initially certain there's an edge, nothing you learn by asking a few questions will change that certainty.

So Szemerédi's Regularity Lemma says that if we ask enough questions about *any* graph—not necessarily an approximable one—we probably end up in a quasirandom rectangle (at which point we might as well stop asking questions).

So Szemerédi's Regularity Lemma says that if we ask enough questions about *any* graph—not necessarily an approximable one—we probably end up in a quasirandom rectangle (at which point we might as well stop asking questions).

Szemerédi's Regularity Lemma became an important tool in parts of graph theory, but it had some limitations: the number of questions you need to ask is huge (a tower of exponents, $2^{2^{\cdot^{\cdot^{\cdot^{2}}}}}$, that gets taller as you ask for that "probably end up" to get closer to probability 1), and you only know that you're *probably* in a quasirandom rectangle.

So Szemerédi's Regularity Lemma says that if we ask enough questions about *any* graph—not necessarily an approximable one—we probably end up in a quasirandom rectangle (at which point we might as well stop asking questions).

Szemerédi's Regularity Lemma became an important tool in parts of graph theory, but it had some limitations: the number of questions you need to ask is huge (a tower of exponents, $2^{2^{\cdot^{\cdot^{\cdot^{2}}}}}$, that gets taller as you ask for that "probably end up" to get closer to probability 1), and you only know that you're *probably* in a quasirandom rectangle.

Tame graph regularity was motivated by showing that for "tame graphs"—graphs which fell on the nice side of a dividing line—these limitations went away, and that for "wild graphs"—graphs on the complex side of a dividing line—these limitations were unavoidable.

Tame Hypergraph Regularity: A new game

The step that gets us to talking about hypergraphs is a simple one: what if, instead of having sets X and Y and talking about whether a pair x, y is an edge,

The step that gets us to talking about hypergraphs is a simple one: what if, instead of having sets X and Y and talking about whether a pair x, y is an edge, we have sets X, Y, and Z, and we talk about whether a triple x, y, z is an edge?

The step that gets us to talking about hypergraphs is a simple one: what if, instead of having sets X and Y and talking about whether a pair x, y is an edge, we have sets X, Y, and Z, and we talk about whether a triple x, y, z is an edge?

We don't know as much about what the dividing lines are for sets of triples. So the goal in tame hypergraph regularity is to find them:

The step that gets us to talking about hypergraphs is a simple one: what if, instead of having sets X and Y and talking about whether a pair x, y is an edge, we have sets X, Y, and Z, and we talk about whether a triple x, y, z is an edge?

We don't know as much about what the dividing lines are for sets of triples. So the goal in tame hypergraph regularity is to find them: we'll figure out what the right analogs of things like approximability and admiting unary error are, and then use those to help identify the dividing lines.

Here are some examples of rules for triples. In all cases, X, Y, and Z are the set of integers up to n.

Here are some examples of rules for triples. In all cases, X, Y, and Z are the set of integers up to n.

- $\mathcal{R}^3_{min}$: x, y, z is an edge when $\min\{x, y\} < z$.

Here are some examples of rules for triples. In all cases, X, Y, and Z are the set of integers up to n.

- $\mathcal{R}^3_{min}$: x, y, z is an edge when $\min\{x, y\} < z$.
- $\mathcal{R}^3_{sum}$: x, y, z is an edge when $x + y + z < n$. (This is actually pretty analogous to our $x < y$ example—if you flip the y axis, $x < y$ is a lot like $x + y < n$.)

Here are some examples of rules for triples. In all cases, X, Y, and Z are the set of integers up to n.

- $\mathcal{R}^3_{min}$: x, y, z is an edge when $\min\{x, y\} < z$.
- $\mathcal{R}^3_{sum}$: x, y, z is an edge when $x + y + z < n$. (This is actually pretty analogous to our $x < y$ example—if you flip the y axis, $x < y$ is a lot like $x + y < n$.)
- $\mathcal{R}^3_{rand}$: We flip a coin for each triple and record the result in a big table.

Here are some examples of rules for triples. In all cases, X, Y, and Z are the set of integers up to n.

- $\mathcal{R}^3_{min}$: x, y, z is an edge when $\min\{x, y\} < z$.
- $\mathcal{R}^3_{sum}$: x, y, z is an edge when $x + y + z < n$. (This is actually pretty analogous to our $x < y$ example—if you flip the y axis, $x < y$ is a lot like $x + y < n$.)
- $\mathcal{R}^3_{rand}$: We flip a coin for each triple and record the result in a big table.
- $\mathcal{R}^3_{rand\ pair}$ We flip a coin for each *pair* and record the result in a big table. x, y, z is an edge if an odd number of the pairs (x, y), (x, z), and (y, z) came up heads.

The last example, $\mathcal{R}^3_{rand\ pair}$, where we flip coins for pairs, illustrates the way the setting with triples is more complicated.

The last example, $\mathcal{R}^3_{rand\ pair}$, where we flip coins for pairs, illustrates the way the setting with triples is more complicated.

If we recruit a friend and we each ask about one variable—you ask about x, I ask about y, our friend asks about z—this example won't be approximable.

The last example, $\mathcal{R}^3_{rand\ pair}$, where we flip coins for pairs, illustrates the way the setting with triples is more complicated.

If we recruit a friend and we each ask about one variable—you ask about x, I ask about y, our friend asks about z—this example won't be approximable.

But it would still be a meaningful game if we were each allowed to ask questions about one of the pairs: you could ask about x and y together, I could ask about x and z together, and our friend could ask about y and z together.

The last example, $\mathcal{R}^3_{rand\ pair}$, where we flip coins for pairs, illustrates the way the setting with triples is more complicated.

If we recruit a friend and we each ask about one variable—you ask about x, I ask about y, our friend asks about z—this example won't be approximable.

But it would still be a meaningful game if we were each allowed to ask questions about one of the pairs: you could ask about x and y together, I could ask about x and z together, and our friend could ask about y and z together. It's still meaningful because none of us can ask about all three at once, so no one can just ask "is the triple an edge?"

So there are (at least) two different versions of our game, a *unary* version where we ask questions about one value at a time, So there are (at least) two different versions of our game, a *unary* version where we ask questions about one value at a time, and a *binary* version where we get to ask questions about two values at a time.

So there are (at least) two different versions of our game, a *unary* version where we ask questions about one value at a time, and a *binary* version where we get to ask questions about two values at a time.

Within each of those versions, we have further variations based on what kind of error they admit: that is, whether we can identify the tricky cases where we aren't sure about the answer with less information than we need to actually determine the answer.

Here's a summary of what's been discovered:

Here's a summary of what's been discovered:

- There are two notions of "approximable", one for each version of the game, and we understand them pretty well.

Here's a summary of what's been discovered:

- There are two notions of "approximable", one for each version of the game, and we understand them pretty well.
- There are notions of "admitting unary error" and "admitting binary error". One we sort-of understand, but it's more complicated that we expected, and the other we really don't understand yet.

Here's a summary of what's been discovered:

- There are two notions of "approximable", one for each version of the game, and we understand them pretty well.
- There are notions of "admitting unary error" and "admitting binary error". One we sort-of understand, but it's more complicated that we expected, and the other we really don't understand yet.
- There's a totally new notion that mixes admitting error coming from unary questions with asking binary questions.

The picture so far looks like this:

Before we investigate those, let's get our terminology right.

Before we investigate those, let's get our terminology right.

Definition

A 3-*graph* consists of three sets, X, Y, and Z, and a set E of triples. We call E the *edges* of the 3-graph.

Technically this might be called a "tripartite 3-graph". Sometimes these are also called *hypergraphs* or 3-*regular hypergraphs*. What we're calling edges might be called *hyperedges* or 3-*edges*.

Before we investigate those, let's get our terminology right.

Definition

A 3-*graph* consists of three sets, X, Y, and Z, and a set E of triples. We call E the *edges* of the 3-graph.

Technically this might be called a "tripartite 3-graph". Sometimes these are also called *hypergraphs* or 3-*regular hypergraphs*. What we're calling edges might be called *hyperedges* or 3-*edges*.

Definition

A *rule for triples* is a set of 3-graphs.

We'll always name rules for triples with a superscript, like $\mathcal{R}^3$, so we don't confuse them with rules for pairs.

Tame Hypergraph Regularity: Approximable rules

We'll start with the notions of approximability.

We'll start with the notions of approximability.

The simplest notion to talk about is being approximable in the unary version of the game, where questions can only be about a single value—about x, or about y, or about z, but not about more than one at a time.

We'll start with the notions of approximability.

The simplest notion to talk about is being approximable in the unary version of the game, where questions can only be about a single value—about x, or about y, or about z, but not about more than one at a time.

Definition

A rule for triples, $\mathcal{R}$, is *unary approximable* if for every $\epsilon > 0$, there is a number N so that for any 3-graph (X, Y, Z, E) in $\mathcal{R}$, if $x \in X$, $y \in Y$, $z \in Z$ are chosen randomly and we ask N questions about each of x, y, and z separately, we can guess whether the triple is an edge and get it right at least $1 - \epsilon$ of the time.

Definition

A rule for triples $\mathcal{R}$ is *unary approximable* if for every $\epsilon > 0$, there is a number N so that for any 3-graph (X, Y, Z, E) in $\mathcal{R}$ and any weights on X, Y, Z, if $x \in X$, $y \in Y$, $z \in Z$ are chosen randomly according to the weights and we ask N questions about each of x, y, and z separately, we can guess whether the triple is an edge and get it right at least $1 - \epsilon$ of the time.

That is, a rule is unary approximable if, when you, me, and a friend team up and we each ask questions about one value, we're usually able to guess if the triple is an edge by combining the information we learned.

Consider the rule $\mathcal{R}^3_{sum}$ where x, y, z is an edge when $x + y + z < n$.

Consider the rule $\mathcal{R}^3_{sum}$ where x, y, z is an edge when $x + y + z < n$.

This is unary approximable: by asking a few questions, we can each pin the value we're asking about down to inside an interval of size $n/8$.

Consider the rule $\mathcal{R}^3_{sum}$ where x, y, z is an edge when $x + y + z < n$.

This is unary approximable: by asking a few questions, we can each pin the value we're asking about down to inside an interval of size $n/8$. For instance, if you learn that $x < n/8$, I learn that $n/8 \leq y < n/4$, and our friend learns that $n/4 \leq z < 3n/8$, we can conclude that $x + y + z$ is definitely less than n.

Consider the rule $\mathcal{R}^3_{sum}$ where x, y, z is an edge when $x + y + z < n$.

This is unary approximable: by asking a few questions, we can each pin the value we're asking about down to inside an interval of size $n/8$. For instance, if you learn that $x < n/8$, I learn that $n/8 \leq y < n/4$, and our friend learns that $n/4 \leq z < 3n/8$, we can conclude that $x + y + z$ is definitely less than n. If you learn that $3n/8 \leq x < n/2$, I learn that $n/2 \leq y < 5n/8$, and our friend learns that $5n/8 \leq z < 3n/4$, we can conclude that $x + y + z$ is definitely bigger than n.

Consider the rule $\mathcal{R}^3_{sum}$ where x, y, z is an edge when $x + y + z < n$.

This is unary approximable: by asking a few questions, we can each pin the value we're asking about down to inside an interval of size $n/8$. For instance, if you learn that $x < n/8$, I learn that $n/8 \leq y < n/4$, and our friend learns that $n/4 \leq z < 3n/8$, we can conclude that $x + y + z$ is definitely less than n. If you learn that $3n/8 \leq x < n/2$, I learn that $n/2 \leq y < 5n/8$, and our friend learns that $5n/8 \leq z < 3n/4$, we can conclude that $x + y + z$ is definitely bigger than n.

A small fraction of the time, we'll be unsure: maybe you and I learn that $n/4 \leq x, y < 3n/8$ while our friend learns that $3n/8 \leq z < n/2$. Then all we know is that $x + y + z$ is between $7n/8$ and $5n/4$, so we know whether $x + y + z < n$.

Consider the rule $\mathcal{R}^3_{sum}$ where x, y, z is an edge when $x + y + z < n$.

This is unary approximable: by asking a few questions, we can each pin the value we're asking about down to inside an interval of size $n/8$. For instance, if you learn that $x < n/8$, I learn that $n/8 \leq y < n/4$, and our friend learns that $n/4 \leq z < 3n/8$, we can conclude that $x + y + z$ is definitely less than n. If you learn that $3n/8 \leq x < n/2$, I learn that $n/2 \leq y < 5n/8$, and our friend learns that $5n/8 \leq z < 3n/4$, we can conclude that $x + y + z$ is definitely bigger than n.

A small fraction of the time, we'll be unsure: maybe you and I learn that $n/4 \leq x, y < 3n/8$ while our friend learns that $3n/8 \leq z < n/2$. Then all we know is that $x + y + z$ is between $7n/8$ and $5n/4$, so we know whether $x + y + z < n$.

But the times we're unsure are a small fraction, so $\mathcal{R}^3_{sum}$ is unary approximable.

Consider the rule $\mathcal{R}^3_{min}$ where x, y, z is an edge when $\min\{x, y\} < z$.

Consider the rule $\mathcal{R}^3_{min}$ where x, y, z is an edge when $\min\{x, y\} < z$.

This rule is also unary approximable, for basically the same reason:

Consider the rule $\mathcal{R}^3_{min}$ where x, y, z is an edge when $\min\{x, y\} < z$.

This rule is also unary approximable, for basically the same reason: each player pins their value to a small interval, and most of the time that's enough to be certain of the comparison.

None of the rules where we make random tables of pairs or triples are going to be unary approximable.

None of the rules where we make random tables of pairs or triples are going to be unary approximable.

If our rule is $\mathcal{R}^3_{rand}$, so we flip a coin for each triple to making a random table, nothing we pin down about x, y, or z separately is enough to help.

None of the rules where we make random tables of pairs or triples are going to be unary approximable.

If our rule is $\mathcal{R}^3_{rand}$, so we flip a coin for each triple to making a random table, nothing we pin down about x, y, or z separately is enough to help.

If our rule is $\mathcal{R}^3_{rand\ pair}$, so we flip a coin for each pair and decide a triple is an edge when an odd number of the pairs got heads, it's still not unary approximable:

None of the rules where we make random tables of pairs or triples are going to be unary approximable.

If our rule is $\mathcal{R}^3_{rand}$, so we flip a coin for each triple to making a random table, nothing we pin down about x, y, or z separately is enough to help.

If our rule is $\mathcal{R}^3_{rand\ pair}$, so we flip a coin for each pair and decide a triple is an edge when an odd number of the pairs got heads, it's still not unary approximable: for instance, nothing we ask about x and y helps much for figuring out if our coin came up heads for the pair x, y.

Here's another example, $\mathcal{R}^3_{fun}$. We'll let X and Y be the numbers up to n, but Z will be the set of *functions* from X to Y, and x, y, z is an edge when $z(x) < y$.

Here's another example, $\mathcal{R}^3_{fun}$. We'll let X and Y be the numbers up to n, but Z will be the set of *functions* from X to Y, and x, y, z is an edge when $z(x) < y$.

This is also not unary approximable.

Here's another example, $\mathcal{R}^3_{fun}$. We'll let X and Y be the numbers up to n, but Z will be the set of *functions* from X to Y, and x, y, z is an edge when $z(x) < y$.

This is also not unary approximable. It's a bit harder to see why, but a good start is trying to come up with questions our friend could be asking about z, and noticing that unless z happens to be a really nice function, you can't learn much about $z(x)$ by asking about x and about z separately.

Just like we saw for graphs, whether a rule for triples is unary approximable is going to be related to whether it has finite VC dimension.

Just like we saw for graphs, whether a rule for triples is unary approximable is going to be related to whether it has finite VC dimension.

What does it mean for a set of *triples* to have finite VC dimension?

Just like we saw for graphs, whether a rule for triples is unary approximable is going to be related to whether it has finite VC dimension.

What does it mean for a set of *triples* to have finite VC dimension? We need to turn our set of triples into a set of pairs. It turns out there are two natural ways to do this. We'll need the other later, but the one we need now comes from looking at "slices".

Just like we saw for graphs, whether a rule for triples is unary approximable is going to be related to whether it has finite VC dimension.

What does it mean for a set of *triples* to have finite VC dimension? We need to turn our set of triples into a set of pairs. It turns out there are two natural ways to do this. We'll need the other later, but the one we need now comes from looking at "slices".

Definition

When E is a set of triples, for any $u \in X \cup Y \cup Z$, the *slice* corresponding to u, E_u, is the set of pairs (v, w) so that $(u, v, w) \in E$.

Just like we saw for graphs, whether a rule for triples is unary approximable is going to be related to whether it has finite VC dimension.

What does it mean for a set of *triples* to have finite VC dimension? We need to turn our set of triples into a set of pairs. It turns out there are two natural ways to do this. We'll need the other later, but the one we need now comes from looking at "slices".

Definition

When E is a set of triples, for any $u \in X \cup Y \cup Z$, the *slice* corresponding to u, E_u, is the set of pairs (v, w) so that $(u, v, w) \in E$.

That is, we fix any one of the values, and ask which pairs give us an edge when taken together with our fixed value.

Just like we saw for graphs, whether a rule for triples is unary approximable is going to be related to whether it has finite VC dimension.

What does it mean for a set of *triples* to have finite VC dimension? We need to turn our set of triples into a set of pairs. It turns out there are two natural ways to do this. We'll need the other later, but the one we need now comes from looking at "slices".

Definition

When E is a set of triples, for any $u \in X \cup Y \cup Z$, the *slice* corresponding to u, E_u, is the set of pairs (v, w) so that $(u, v, w) \in E$.

That is, we fix any one of the values, and ask which pairs give us an edge when taken together with our fixed value. I'm using the letters u, v, w to emphasize that this is symmetric: we can fix an X value and look at pairs from $Y \times Z$, or fix a Y value and look at pairs from $X \times Z$, or fix a Z value and look at pairs from $Y \times Z$.

Definition

When $\mathcal{R}^3$ is a rule for triples, $\mathcal{R}^3$ has *finite slicewise VC dimension* if there is an n so that, for every (X, Y, Z, E) in $\mathcal{R}^3$ and every $u \in X \cup Y \cup Z$, E_u omits some $n \times n$ subgraph.

Definition

When $\mathcal{R}^3$ is a rule for triples, $\mathcal{R}^3$ has *finite slicewise VC dimension* if there is an n so that, for every (X, Y, Z, E) in $\mathcal{R}^3$ and every $u \in X \cup Y \cup Z$, E_u omits some $n \times n$ subgraph.

Another way to say this is that a rule for triples gives us a rule for pairs—take *every* slice from *every* 3-graph in $\mathcal{R}^3$.

Definition

When $\mathcal{R}^3$ is a rule for triples, $\mathcal{R}^3$ has *finite slicewise VC dimension* if there is an n so that, for every (X, Y, Z, E) in $\mathcal{R}^3$ and every $u \in X \cup Y \cup Z$, E_u omits some $n \times n$ subgraph.

Another way to say this is that a rule for triples gives us a rule for pairs—take *every* slice from *every* 3-graph in $\mathcal{R}^3$. Saying $\mathcal{R}^3$ has finite slicewise VC dimension us exactly saying that this rule consisting of all the slices has finite VC dimension.

For example, the slices of $\mathcal{R}^3_{sum}$, where x, y, z is an edge when $x + y + z < n$, all look similar:

For example, the slices of $\mathcal{R}^3_{sum}$, where x, y, z is an edge when $x + y + z < n$, all look similar: the slices are all the pairs v, w where $v + w < k$ for some k.

For example, the slices of $\mathcal{R}^3_{sum}$, where x, y, z is an edge when $x + y + z < n$, all look similar: the slices are all the pairs v, w where $v + w < k$ for some k.

This is basically an example we've already seen: it omits ▚.

For example, the slices of $\mathcal{R}^3_{sum}$, where x, y, z is an edge when $x + y + z < n$, all look similar: the slices are all the pairs v, w where $v + w < k$ for some k.

This is basically an example we've already seen: it omits . Because in order to get this subgraph, we would need two rows (say, v and v') and two columns (say, w and w') so that $v + w < k$, $v + w' \geq k$, $v' + w \geq k$, $v' + w' < k$.

For example, the slices of $\mathcal{R}^3_{sum}$, where x, y, z is an edge when $x + y + z < n$, all look similar: the slices are all the pairs v, w where $v + w < k$ for some k.

This is basically an example we've already seen: it omits . Because in order to get this subgraph, we would need two rows (say, v and v') and two columns (say, w and w') so that $v + w < k$, $v + w' \geq k$, $v' + w \geq k$, $v' + w' < k$. But this is impossible: $v + w < k$ and $v + w' \geq k$ means that $w < w'$.

For example, the slices of $\mathcal{R}^3_{sum}$, where x, y, z is an edge when $x + y + z < n$, all look similar: the slices are all the pairs v, w where $v + w < k$ for some k.

This is basically an example we've already seen: it omits . Because in order to get this subgraph, we would need two rows (say, v and v') and two columns (say, w and w') so that $v + w < k$, $v + w' \geq k$, $v' + w \geq k$, $v' + w' < k$. But this is impossible: $v + w < k$ and $v + w' \geq k$ means that $w < w'$. Dually, $v' + w \geq k$ while $v' + w' < k$ means that $w' < w$. These can't both be true.

On the other hand, the example $\mathcal{R}^3_{fun}$, where Z is the functions from X to Y and the edges are where $z(x) < y$, does not have finite slicewise VC dimension.

On the other hand, the example $\mathcal{R}^3_{fun}$, where Z is the functions from X to Y and the edges are where $z(x) < y$, does not have finite slicewise VC dimension.

Let's show that there's a slice which contains every $n \times n$ graph.

On the other hand, the example $\mathcal{R}^3_{fun}$, where Z is the functions from X to Y and the edges are where $z(x) < y$, does not have finite slicewise VC dimension.

Let's show that there's a slice which contains every $n \times n$ graph. Let X and Y be the numbers less than $n+1$ and pick any value of y other than 1. Pick your favorite $n \times n$ graph.

On the other hand, the example $\mathcal{R}^3_{fun}$, where Z is the functions from X to Y and the edges are where $z(x) < y$, does not have finite slicewise VC dimension.

Let's show that there's a slice which contains every $n \times n$ graph. Let X and Y be the numbers less than $n+1$ and pick any value of y other than 1. Pick your favorite $n \times n$ graph.

We'll let the n values in X, 1 through n, be our columns.

On the other hand, the example $\mathcal{R}^3_{fun}$, where Z is the functions from X to Y and the edges are where $z(x) < y$, does not have finite slicewise VC dimension.

Let's show that there's a slice which contains every $n \times n$ graph. Let X and Y be the numbers less than $n+1$ and pick any value of y other than 1. Pick your favorite $n \times n$ graph.

We'll let the n values in X, 1 through n, be our columns. Then for each row of our graph, choose a z which gives exactly that row: to make the j-th row, choose z_j so that $z_j(i) = 1$ when (i,j) is an edge and $z_j(i) = y$ when (i,j) is not an edge.

On the other hand, the example $\mathcal{R}^3_{fun}$, where Z is the functions from X to Y and the edges are where $z(x) < y$, does not have finite slicewise VC dimension.

Let's show that there's a slice which contains every $n \times n$ graph. Let X and Y be the numbers less than $n+1$ and pick any value of y other than 1. Pick your favorite $n \times n$ graph.

We'll let the n values in X, 1 through n, be our columns. Then for each row of our graph, choose a z which gives exactly that row: to make the j-th row, choose z_j so that $z_j(i) = 1$ when (i, j) is an edge and $z_j(i) = y$ when (i, j) is not an edge.

Then $1, \dots, n$ and $z_1, \dots, z_n$ are a copy of the graph we wanted. Since we can do that for any graph, E_y contains every $n \times n$ graph.

As those examples suggest, slicewise VC dimension is the notion we need to characterize unary approximations.

As those examples suggest, slicewise VC dimension is the notion we need to characterize unary approximations.

Theorem

A rule for triples is unary approximable if and only if it has finite slicewise VC dimension.

What about the other version of the game, where we get to ask questions about two values at once?

What about the other version of the game, where we get to ask questions about two values at once?

Definition

A rule for triples, $\mathcal{R}$, is *binary approximable* if for every $\epsilon > 0$, there is a number N so that for any 3-graph (X, Y, Z, E) in $\mathcal{R}$, if $x \in X$, $y \in Y$, $z \in Z$ are chosen randomly and we ask N questions about each of the pairs (x, y), (x, z), and (y, z) separately, we can guess whether the triple is an edge and get it right at least $1 - \epsilon$ of the time.

It’s easier for us to win the binary game than the unary game, It's easier for us to win the binary game than the unary game, because when you get to ask questions about the pair (x, y), you could choose to only ask about x, and similarly for the other two players.

It's easier for us to win the binary game than the unary game, because when you get to ask questions about the pair (x, y), you could choose to only ask about x, and similarly for the other two players. The binary game just gives us more options, and therefore more rules will be binary approximable.

For instance, recall $\mathcal{R}^3_{rand\ pair}$, the rule where we make a random table by flipping a coin for each pair, and x, y, z is an edge when an odd number of the pairs got heads.

For instance, recall $\mathcal{R}^3_{rand\ pair}$, the rule where we make a random table by flipping a coin for each pair, and x, y, z is an edge when an odd number of the pairs got heads.

That isn't unary approximable, but it's easy to give a binary approximation,

For instance, recall $\mathcal{R}^3_{rand\ pair}$, the rule where we make a random table by flipping a coin for each pair, and x, y, z is an edge when an odd number of the pairs got heads.

That isn't unary approximable, but it's easy to give a binary approximation, since we can each just outright ask whether the coin flipped for our pair came up heads.

Similarly, with the rule $\mathcal{R}^3_{fun}$, where Z is the functions from X to Y and x, y, z is an edge when $z(x) < y$,

Similarly, with the rule $\mathcal{R}^3_{fun}$, where Z is the functions from X to Y and x, y, z is an edge when $z(x) < y$, I'll use my questions to ask about the value of $z(x)$ in order to pin it down to a small interval.

Similarly, with the rule $\mathcal{R}^3_{fun}$, where Z is the functions from X to Y and x, y, z is an edge when $z(x) < y$, I'll use my questions to ask about the value of $z(x)$ in order to pin it down to a small interval.

One of the other players asks questions about y to pin y down to a small interval as well.

Similarly, with the rule $\mathcal{R}^3_{fun}$, where Z is the functions from X to Y and x, y, z is an edge when $z(x) < y$, I'll use my questions to ask about the value of $z(x)$ in order to pin it down to a small interval.

One of the other players asks questions about y to pin y down to a small interval as well. Then we compare our answers and most of the time the intervals don't overlap, so we know for sure whether $z(x) < y$.

When the rule is $\mathcal{R}^3_{rand}$, where we flip a coin for each *triple*, though, asking questions about pairs doesn't help.

When the rule is $\mathcal{R}^3_{rand}$, where we flip a coin for each *triple*, though, asking questions about pairs doesn't help.

This rule isn't binary approximable: no matter how many questions about pairs we ask, it doesn't help us figure out if the coin came up heads for the particular triple we're dealing with.

Finite VC dimension asked about omitting graphs.

Finite VC dimension asked about omitting graphs. Stated like that, we might guess how to generalize to 3-graphs:

Finite VC dimension asked about omitting graphs. Stated like that, we might guess how to generalize to 3-graphs:

Definition
A rule for triples $\mathcal{R}$ has *finite VC_2 dimension* if there is some n so that every 3-graph in $\mathcal{R}$ omits some $n \times n \times n$ sub-3-graph.

Consider the rule $\mathcal{R}^3_{rand\ pair}$ where we flip a coin for each pair, and x, y, z is an edge if an odd number of the coins for pairs came up heads.

Consider the rule $\mathcal{R}^3_{rand\ pair}$ where we flip a coin for each pair, and x, y, z is an edge if an odd number of the coins for pairs came up heads.

This has finite VC_2 dimension: it omits any $2 \times 2 \times 2$ 3-graph in which exactly 7 out of the 8 possible edges are present.

Consider the rule $\mathcal{R}^3_{rand\ pair}$ where we flip a coin for each pair, and x, y, z is an edge if an odd number of the coins for pairs came up heads.

This has finite VC_2 dimension: it omits any $2 \times 2 \times 2$ 3-graph in which exactly 7 out of the 8 possible edges are present.

For suppose we had a copy of this 3-graph—values x, x', y, y', and z, z' so that all combinations other than x', y', z' are edges.

Consider the rule $\mathcal{R}^3_{rand\ pair}$ where we flip a coin for each pair, and x, y, z is an edge if an odd number of the coins for pairs came up heads.

This has finite VC_2 dimension: it omits any $2 \times 2 \times 2$ 3-graph in which exactly 7 out of the 8 possible edges are present.

For suppose we had a copy of this 3-graph—values x, x', y, y', and z, z' so that all combinations other than x', y', z' are edges. Let us write $c_{x,y}$ for the number which is 1 if the coin for x, y came up heads, and 0 if it came up tails, and similarly for the other pairs.

Consider the rule $\mathcal{R}^3_{rand\ pair}$ where we flip a coin for each pair, and x, y, z is an edge if an odd number of the coins for pairs came up heads.

This has finite VC_2 dimension: it omits any $2 \times 2 \times 2$ 3-graph in which exactly 7 out of the 8 possible edges are present.

For suppose we had a copy of this 3-graph—values x, x', y, y', and z, z' so that all combinations other than x', y', z' are edges. Let us write $c_{x,y}$ for the number which is 1 if the coin for x, y came up heads, and 0 if it came up tails, and similarly for the other pairs.

So $c_{x,y} + c_{x,z} + c_{y,z}$ must be odd while $c_{x',y'} + c_{x',z'} + c_{y',z'}$ must be even.

If we add up the seven combinations we know are odd, we get

$$\begin{aligned}&(c_{x,y}+c_{x,z}+c_{y,z})+(c_{x',y}+c_{x',z}+c_{y,z})+(c_{x,y'}+c_{x,z}+c_{y',z})\\+&(c_{x,y}+c_{x,z'}+c_{y,z'})+(c_{x',y'}+c_{x',z}+c_{y',z})+(c_{x,y'}+c_{x,z'}+c_{y',z'})\\+&(c_{x',y}+c_{x',z'}+c_{y,z'})\end{aligned}$$

which is a sum of 7 odd numbers, so also odd.

If we add up the seven combinations we know are odd, we get

$$\begin{aligned}
&(c_{x,y}+c_{x,z}+c_{y,z})+(c_{x',y}+c_{x',z}+c_{y,z})+(c_{x,y'}+c_{x,z}+c_{y',z})\\
&+(c_{x,y}+c_{x,z'}+c_{y,z'})+(c_{x',y'}+c_{x',z}+c_{y',z})+(c_{x,y'}+c_{x,z'}+c_{y',z'})\\
&+(c_{x',y}+c_{x',z'}+c_{y,z'})
\end{aligned}$$

which is a sum of 7 odd numbers, so also odd.

Whenever a number appears twice, let's cancel it out—either they're both 0, or they're both odd, so either way they don't affect whether the sum is even or odd. We're left with:

If we add up the seven combinations we know are odd, we get

$$\begin{aligned}&(c_{x,y}+c_{x,z}+c_{y,z})+(c_{x',y}+c_{x',z}+c_{y,z})+(c_{x,y'}+c_{x,z}+c_{y',z})\\+&(c_{x,y}+c_{x,z'}+c_{y,z'})+(c_{x',y'}+c_{x',z}+c_{y',z})+(c_{x,y'}+c_{x,z'}+c_{y',z'})\\+&(c_{x',y}+c_{x',z'}+c_{y,z'})\end{aligned}$$

which is a sum of 7 odd numbers, so also odd.

Whenever a number appears twice, let's cancel it out—either they're both 0, or they're both odd, so either way they don't affect whether the sum is even or odd. We're left with:

$$c_{x',y'}+c_{x',z'}+c_{y',z'}$$

which must also be odd.

If we add up the seven combinations we know are odd, we get

$$\begin{aligned}&(c_{x,y}+c_{x,z}+c_{y,z})+(c_{x',y}+c_{x',z}+c_{y,z})+(c_{x,y'}+c_{x,z}+c_{y',z})\\ +&(c_{x,y}+c_{x,z'}+c_{y,z'})+(c_{x',y'}+c_{x',z}+c_{y',z})+(c_{x,y'}+c_{x,z'}+c_{y',z'})\\ +&(c_{x',y}+c_{x',z'}+c_{y,z'})\end{aligned}$$

which is a sum of 7 odd numbers, so also odd.

Whenever a number appears twice, let's cancel it out—either they're both 0, or they're both odd, so either way they don't affect whether the sum is even or odd. We're left with:

$$c_{x',y'}+c_{x',z'}+c_{y',z'}$$

which must also be odd. But this exactly tells us that an odd number of the pairs (x',y'), (x',z'), and (y',z') must have come up heads:

If we add up the seven combinations we know are odd, we get

$$\begin{aligned}&(c_{x,y}+c_{x,z}+c_{y,z})+(c_{x',y}+c_{x',z}+c_{y,z})+(c_{x,y'}+c_{x,z}+c_{y',z})\\&+(c_{x,y}+c_{x,z'}+c_{y,z'})+(c_{x',y'}+c_{x',z}+c_{y',z})+(c_{x,y'}+c_{x,z'}+c_{y',z'})\\&+(c_{x',y}+c_{x',z'}+c_{y,z'})\end{aligned}$$

which is a sum of 7 odd numbers, so also odd.

Whenever a number appears twice, let's cancel it out—either they're both 0, or they're both odd, so either way they don't affect whether the sum is even or odd. We're left with:

$$c_{x',y'}+c_{x',z'}+c_{y',z'}$$

which must also be odd. But this exactly tells us that an odd number of the pairs (x',y'), (x',z'), and (y',z') must have come up heads: if 7 of the 8 edges are present, the eighth must be as well.

Finite VC_2 dimension gives the right notion for binary approximations.

Finite VC_2 dimension gives the right notion for binary approximations.

Theorem (Chernikov-Towsner)

A rule for triples is binary approximable if and only if it has finite VC_2 *dimension.*

Finite VC_2 dimension gives the right notion for binary approximations.

Theorem (Chernikov-Towsner)

A rule for triples is binary approximable if and only if it has finite VC_2 dimension.

This wasn't the first result about tame hypergraph regularity, but it was the first result that came from the perspective here: we had conjectured that *something* should be equivalent to binary approximable, guessed that it would be finite VC_2 dimension, and then looked for the proof.

Tame Hypergraph Regularity: Admitting unary error

We can just combine our definitions to guess what admitting unary error should be:

We can just combine our definitions to guess what admitting unary error should be:

We will say that $\mathcal{R}^3$ *admits unary error* if, for any $\epsilon > 0$, there is an N so that after we have each asked N questions about one of the variables, we are each allowed to give up ϵ of the time, and any time none of us gives up, we are able to get the right answer by combining our information.

We can just combine our definitions to guess what admitting unary error should be:

We will say that $\mathcal{R}^3$ *admits unary error* if, for any $\epsilon > 0$, there is an N so that after we have each asked N questions about one of the variables, we are each allowed to give up ϵ of the time, and any time none of us gives up, we are able to get the right answer by combining our information.

Once again, the exact definition gives us slightly more leeway to get the answer wrong, but we'll ignore this extra complication.

Based on what happened for just approximability, there's a clear first guess for what dividing line to look for.

Based on what happened for just approximability, there's a clear first guess for what dividing line to look for.

Definition
When $\mathcal{R}^3$ is a rule for triples, $\mathcal{R}^3$ is *slicewise stable* if there is a k so that, for every (X, Y, Z, E) in $\mathcal{R}^3$ and every $u \in X \cup Y \cup Z$, E_u is k-stable.

Based on what happened for just approximability, there's a clear first guess for what dividing line to look for.

Definition

When $\mathcal{R}^3$ is a rule for triples, $\mathcal{R}^3$ is *slicewise stable* if there is a k so that, for every (X, Y, Z, E) in $\mathcal{R}^3$ and every $u \in X \cup Y \cup Z$, E_u is k-stable.

One implication works, using basically the same arguments that worked for pairs.

Theorem

If $\mathcal{R}^3$ admits unary error then $\mathcal{R}^3$ is slicewise stable.

The other direction was open for a little while, and I finally found a counterexample.

The other direction was open for a little while, and I finally found a counterexample.

Consider the following rule for triples, $\mathcal{R}^3_{trinary}$. X, Y, and Z will be sequences of length n of digits $\{0, 1, 2\}$.

The other direction was open for a little while, and I finally found a counterexample.

Consider the following rule for triples, $\mathcal{R}^3_{trinary}$. X, Y, and Z will be sequences of length n of digits $\{0, 1, 2\}$. In order for x, y, z to be an edge, they need to not be all three the same sequence, and at the first digit where they're not all the same, they need to have all three different values.

The other direction was open for a little while, and I finally found a counterexample.

Consider the following rule for triples, $\mathcal{R}^3_{trinary}$. X, Y, and Z will be sequences of length n of digits $\{0, 1, 2\}$. In order for x, y, z to be an edge, they need to not be all three the same sequence, and at the first digit where they're not all the same, they need to have all three different values. So $01000, 01100, 01200$ is an edge, while $01000, 01100, 02200$ is not.

The other direction was open for a little while, and I finally found a counterexample.

Consider the following rule for triples, $\mathcal{R}^3_{trinary}$. X, Y, and Z will be sequences of length n of digits $\{0, 1, 2\}$. In order for x, y, z to be an edge, they need to not be all three the same sequence, and at the first digit where they're not all the same, they need to have all three different values. So $01000, 01100, 01200$ is an edge, while $01000, 01100, 02200$ is not.

The slices E_u are all stable—roughly speaking, what happens to a triple u, v, w is determined by the first coordinate at which v or w differs from u, which is too restrictive to be unstable.

X, Y, and Z will be sequences of length n of digits $\{0, 1, 2\}$. In order for x, y, z to be an edge, they need to not be all three the same sequence, and at the first digit where they're not all the same, they need to take have all three different values.

But this example doesn't admit unary error.

X, Y, and Z will be sequences of length n of digits $\{0, 1, 2\}$. In order for x, y, z to be an edge, they need to not be all three the same sequence, and at the first digit where they're not all the same, they need to take have all three different values.

But this example doesn't admit unary error. The idea is that the natural questions for us to ask are about the initial digits of our value: does it start with a 0, is the second digit a 1, and so on.

X, Y, and Z will be sequences of length n of digits $\{0, 1, 2\}$. In order for x, y, z to be an edge, they need to not be all three the same sequence, and at the first digit where they're not all the same, they need to take have all three different values.

But this example doesn't admit unary error. The idea is that the natural questions for us to ask are about the initial digits of our value: does it start with a 0, is the second digit a 1, and so on. But those questions will always leave boxes where x, y, z all share the same initial digits, and no one looking at just one of x, y, or z on its own can tell if that's going to happen.

X, Y, and Z will be sequences of length n of digits $\{0, 1, 2\}$. In order for x, y, z to be an edge, they need to not be all three the same sequence, and at the first digit where they're not all the same, they need to take have all three different values.

But this example doesn't admit unary error. The idea is that the natural questions for us to ask are about the initial digits of our value: does it start with a 0, is the second digit a 1, and so on. But those questions will always leave boxes where x, y, z all share the same initial digits, and no one looking at just one of x, y, or z on its own can tell if that's going to happen.

It's more work, of course, to show that there aren't some cleverer questions which do better.

So slicewise stability isn't enough. But I mentioned earlier that there's a second way to lift properties of graphs to 3-graphs.

So slicewise stability isn't enough. But I mentioned earlier that there's a second way to lift properties of graphs to 3-graphs.

Another way to get graphs from 3-graphs is to view triples as being pairs where one element happens to itself be a pair:

So slicewise stability isn't enough. But I mentioned earlier that there's a second way to lift properties of graphs to 3-graphs.

Another way to get graphs from 3-graphs is to view triples as being pairs where one element happens to itself be a pair: instead of triples from $X \times Y \times Z$, we have pairs from X and $Y \times Z$, or from Y and $X \times Z$, or from Z and $X \times Y$.

So slicewise stability isn't enough. But I mentioned earlier that there's a second way to lift properties of graphs to 3-graphs.

Another way to get graphs from 3-graphs is to view triples as being pairs where one element happens to itself be a pair: instead of triples from $X \times Y \times Z$, we have pairs from X and $Y \times Z$, or from Y and $X \times Z$, or from Z and $X \times Y$.

Definition

A rule for triples, $\mathcal{R}^3$, is *partitionwise stable* if there is a k so that, for every (X, Y, Z, E) in $\mathcal{R}^3$, all three graphs $(X, Y \times Z, E)$, $(Y, X \times Z, E)$, and $(Z, X \times Y, E)$ are k-stable.

So slicewise stability isn't enough. But I mentioned earlier that there's a second way to lift properties of graphs to 3-graphs.

Another way to get graphs from 3-graphs is to view triples as being pairs where one element happens to itself be a pair: instead of triples from $X \times Y \times Z$, we have pairs from X and $Y \times Z$, or from Y and $X \times Z$, or from Z and $X \times Y$.

Definition

A rule for triples, $\mathcal{R}^3$, is *partitionwise stable* if there is a k so that, for every (X, Y, Z, E) in $\mathcal{R}^3$, all three graphs $(X, Y \times Z, E)$, $(Y, X \times Z, E)$, and $(Z, X \times Y, E)$ are k-stable.

Partitionwise stability is a stronger property: any partitionwise stable rule is slicewise stable, but the example above was slicewise stable without being partitionwise stable.

Theorem
$\mathcal{R}^3$ is partitionwise stable exactly when $\mathcal{R}^3$ admits unary error.

Theorem

$\mathcal{R}^3$ is partitionwise stable exactly when $\mathcal{R}^3$ admits unary error.

It turns out that the subtleties we've been ignoring create some complications here: when we get into the details, there are two definitions which are equivalent for graphs, but lead to distinct generalizations for 3-graphs. One is equivalent to partitionwise stability while the other ends up being strictly between partitionwise and slicewise stability.

Tame Hypergraph Regularity: Admitting binary error

The situation for admitting binary error is even messier right now.

The situation for admitting binary error is even messier right now.

We can guess what admitting binary error should mean: $\mathcal{R}$ *admits binary error* if, for any $\epsilon > 0$, there is an N so that after we have each asked N questions about two of the variables, we are each allowed to give up ϵ of the time, and any time none of us gives up, we are able to get the right answer by combining our information.

The situation for admitting binary error is even messier right now.

We can guess what admitting binary error should mean: $\mathcal{R}$ *admits binary error* if, for any $\epsilon > 0$, there is an N so that after we have each asked N questions about two of the variables, we are each allowed to give up ϵ of the time, and any time none of us gives up, we are able to get the right answer by combining our information.

This is exactly like our definition of admitting unary error, except that the players ask questions about two variables at a time.

For graphs, confident approximations were characterized by omitting the half-graphs.

For graphs, confident approximations were characterized by omitting the half-graphs. We know what the analog of the half-graph is for 3-graphs: it's the 3-graphs in $\mathcal{R}^3_{sum}$, where there's an edge when $x + y + z < n$.

For graphs, confident approximations were characterized by omitting the half-graphs. We know what the analog of the half-graph is for 3-graphs: it's the 3-graphs in $\mathcal{R}^3_{sum}$, where there's an edge when $x + y + z < n$.

One way to see the resemblence is to think about the pictures. The half-graph looks like a square divided along the diagonal, while a drawing a of $\mathcal{R}^3_{sum}$ would look like a cube divided along the diagonal.

For graphs, confident approximations were characterized by omitting the half-graphs. We know what the analog of the half-graph is for 3-graphs: it's the 3-graphs in $\mathcal{R}^3_{sum}$, where there's an edge when $x + y + z < n$.

One way to see the resemblence is to think about the pictures. The half-graph looks like a square divided along the diagonal, while a drawing a of $\mathcal{R}^3_{sum}$ would look like a cube divided along the diagonal.

Indeed, $\mathcal{R}^3_{sum}$ does not admit binary error.

But Terry and Wolf found a family of other counterexamples.

But Terry and Wolf found a family of other counterexamples. The first example in this family is called GS_3. In $\mathcal{R}^3_{GS_3}$, the sets $X = Y = Z$ are the set of sequences of digits in $\{0, 1, 2\}$ of length n.

But Terry and Wolf found a family of other counterexamples. The first example in this family is called GS_3. In $\mathcal{R}^3_{GS_3}$, the sets $X = Y = Z$ are the set of sequences of digits in $\{0, 1, 2\}$ of length n.

To figure out if x, y, z is an edge, we add the sequences, position by position, modulo 3—that is, we add up the first digits of each, and if it's more than 3, subtract 3 until we get 0, 1, or 2. Then we add the second digit the same way, and so on.

But Terry and Wolf found a family of other counterexamples. The first example in this family is called GS_3. In $\mathcal{R}^3_{GS_3}$, the sets $X = Y = Z$ are the set of sequences of digits in $\{0, 1, 2\}$ of length n.

To figure out if x, y, z is an edge, we add the sequences, position by position, modulo 3—that is, we add up the first digits of each, and if it's more than 3, subtract 3 until we get 0, 1, or 2. Then we add the second digit the same way, and so on. For instance, if we have the triple 0000001112, 0001121122, 0121221222, adding them digit by digit modulo 3 gives 0122010120.

But Terry and Wolf found a family of other counterexamples.The first example in this family is called GS_3. In $\mathcal{R}^3_{GS_3}$, the sets $X = Y = Z$ are the set of sequences of digits in $\{0, 1, 2\}$ of length n.

To figure out if x, y, z is an edge, we add the sequences, position by position, modulo 3—that is, we add up the first digits of each, and if it's more than 3, subtract 3 until we get 0, 1, or 2. Then we add the second digit the same way, and so on. For instance, if we have the triple 0000001112, 0001121122, 0121221222, adding them digit by digit modulo 3 gives 0122010120.

x, y, z is an edge if the first 1 appears before the first 2.

Showing that $\mathcal{R}^3_{GS_3}$ doesn't admit binary error is quite difficult.

Showing that $\mathcal{R}^3_{GS_3}$ doesn't admit binary error is quite difficult.

Some intuition starts with the observation that if you know x and y, even in full detail, there's always a z which gives us an edge and a z which gives us a non-edge:

Showing that $\mathcal{R}^3_{GS_3}$ doesn't admit binary error is quite difficult.

Some intuition starts with the observation that if you know x and y, even in full detail, there's always a z which gives us an edge and a z which gives us a non-edge: knowing two coordinates, even in their entirety, doesn't help predict what's going to happen.

More generally, instead of having sequences of digits in $\{0, 1, 2\}$, we could have digits $\{0, 1, 2, \ldots, k\}$ for some $k \geq 2$.

More generally, instead of having sequences of digits in $\{0, 1, 2\}$, we could have digits $\{0, 1, 2, \ldots, k\}$ for some $k \geq 2$. We add digits modulo k, and to decide whether x, y, z is an edge, we look at the first non-zero digit, and decide whether it's an edge based on this value—say, a 1 means an edge and any other value means a non-edge.

More generally, instead of having sequences of digits in $\{0, 1, 2\}$, we could have digits $\{0, 1, 2, \ldots, k\}$ for some $k \geq 2$. We add digits modulo k, and to decide whether x, y, z is an edge, we look at the first non-zero digit, and decide whether it's an edge based on this value—say, a 1 means an edge and any other value means a non-edge. All these variants also fail to admit binary error.

Furthermore, $\mathcal{R}^3_{sum}$ and $\mathcal{R}^3_{GS_3}$ are *distinct* obstacles to admitting binary error: neither is contained in the other.

Furthermore, $\mathcal{R}^3_{sum}$ and $\mathcal{R}^3_{GS_3}$ are *distinct* obstacles to admitting binary error: neither is contained in the other.

You might hope that there's some common obstruction contained in both, but there's not.

Furthermore, $\mathcal{R}^3_{sum}$ and $\mathcal{R}^3_{GS_3}$ are *distinct* obstacles to admitting binary error: neither is contained in the other.

You might hope that there's some common obstruction contained in both, but there's not.

Theorem

If $\mathcal{R}^3$ is contained in both $\mathcal{R}^3_{sum}$ and $\mathcal{R}^3_{GS_3}$ then $\mathcal{R}^3$ admits binary error.

So, unlike our other dividing lines, there isn't going to be a single family of 3-graphs which characterize admitting binary error.

So, unlike our other dividing lines, there isn't going to be a single family of 3-graphs which characterize admitting binary error.

There's some hope for a different kind of characterization, perhaps with "partial 3-graphs" (where some triples are not committed to being either edges or non-edges). There is a characterization involving some fairly complicated "trees".

Tame Hypergraph Regularity: Admitting linear error

The notion of admitting binary error is incomparable with being unary approximable—a particular rule could have either one, neither, or both:

The notion of admitting binary error is incomparable with being unary approximable—a particular rule could have either one, neither, or both:

The basic examples of things which don't admit binary error, like $\mathcal{R}^3_{sum}$ and $\mathcal{R}^3_{GS_3}$, are unary approximable.

The notion of admitting binary error is incomparable with being unary approximable—a particular rule could have either one, neither, or both:

The basic examples of things which don't admit binary error, like $\mathcal{R}^3_{sum}$ and $\mathcal{R}^3_{GS_3}$, are unary approximable. And the basic examples of things which are not unary approximable, like $\mathcal{R}^3_{rand\ pair}$, do admit binary error.

The notion of admitting binary error is incomparable with being unary approximable—a particular rule could have either one, neither, or both:

The basic examples of things which don't admit binary error, like $\mathcal{R}^3_{sum}$ and $\mathcal{R}^3_{GS_3}$, are unary approximable. And the basic examples of things which are not unary approximable, like $\mathcal{R}^3_{rand\ pair}$, do admit binary error.

There's a notion that encompasses all these examples, but is still more restrictive than being merely binary approximable.

To explain this final notion, we need a new game which combines our previous games.

To explain this final notion, we need a new game which combines our previous games.

In the new game, we play both versions of the binary game, sequentially:

To explain this final notion, we need a new game which combines our previous games.

In the new game, we play both versions of the binary game, sequentially: first, we play the unary game, each asking questions about one value. Then we get together and share information.

To explain this final notion, we need a new game which combines our previous games.

In the new game, we play both versions of the binary game, sequentially: first, we play the unary game, each asking questions about one value. Then we get together and share information. Then, based on that information, we can ask more questions, this time about two values at time. Then we get together and base our guess on that new information. A small fraction of the time, we're allowed to give up, but most of the time, we have to make a guess and get it correct.

To explain this final notion, we need a new game which combines our previous games.

In the new game, we play both versions of the binary game, sequentially: first, we play the unary game, each asking questions about one value. Then we get together and share information. Then, based on that information, we can ask more questions, this time about two values at time. Then we get together and base our guess on that new information. A small fraction of the time, we're allowed to give up, but most of the time, we have to make a guess and get it correct.

All our variations amount to tweaking two parameters of this general game: which stages we ask questions at, and when we decide whether to give up.

Binary approximable lets us wait until the very end, after we've shared the binary information, to decide whether to give up. (Whether or not we play the unary stage doesn't matter because when we get the full binary stage, it doesn't help us.)

Binary approximable lets us wait until the very end, after we've shared the binary information, to decide whether to give up. (Whether or not we play the unary stage doesn't matter because when we get the full binary stage, it doesn't help us.) Unary approximable corresponds to skipping the second stage, but waiting until the very end, after we share the unary information, to decide whether to give up.

Binary approximable lets us wait until the very end, after we've shared the binary information, to decide whether to give up. (Whether or not we play the unary stage doesn't matter because when we get the full binary stage, it doesn't help us.) Unary approximable corresponds to skipping the second stage, but waiting until the very end, after we share the unary information, to decide whether to give up.

Admitting unary error skips the second stage and requires us to decide whether to give up *before* sharing information, and admitting binary error skips the first stage and requires us to decide whether to give up before sharing information.

Binary approximable lets us wait until the very end, after we've shared the binary information, to decide whether to give up. (Whether or not we play the unary stage doesn't matter because when we get the full binary stage, it doesn't help us.) Unary approximable corresponds to skipping the second stage, but waiting until the very end, after we share the unary information, to decide whether to give up.

Admitting unary error skips the second stage and requires us to decide whether to give up *before* sharing information, and admitting binary error skips the first stage and requires us to decide whether to give up before sharing information.

Our new notion comes from playing both stages, and requiring that we decide whether to give up after sharing the unary information but before asking binary question.

Here's what that means.

Here's what that means. The house will pick an $\epsilon > 0$ and we'll say how many questions we need at each stage. Then the house picks a 3-graph from the rule and weights and chooses x, y, and z.

Here's what that means. The house will pick an $\epsilon > 0$ and we'll say how many questions we need at each stage. Then the house picks a 3-graph from the rule and weights and chooses x, y, and z.

Then we ask our unary questions. When we get back together, a small fraction ($< \epsilon$) of the time, we're allowed to say we don't know what's going to happen.

Here's what that means. The house will pick an $\epsilon > 0$ and we'll say how many questions we need at each stage. Then the house picks a 3-graph from the rule and weights and chooses x, y, and z.

Then we ask our unary questions. When we get back together, a small fraction ($< \epsilon$) of the time, we're allowed to say we don't know what's going to happen. The rest of the time, we go back and ask our binary questions, compare our answers, and make our guess, and now we need to get the right answer.

Here's what that means. The house will pick an $\epsilon > 0$ and we'll say how many questions we need at each stage. Then the house picks a 3-graph from the rule and weights and chooses x, y, and z.

Then we ask our unary questions. When we get back together, a small fraction ($< \epsilon$) of the time, we're allowed to say we don't know what's going to happen. The rest of the time, we go back and ask our binary questions, compare our answers, and make our guess, and now we need to get the right answer.

That is, we don't have to get it right all the time, but we need to decide *at the end of the unary game* whether or not we're going to get it right, before we get to ask any binary questions.

Here's what that means. The house will pick an $\epsilon > 0$ and we'll say how many questions we need at each stage. Then the house picks a 3-graph from the rule and weights and chooses x, y, and z.

Then we ask our unary questions. When we get back together, a small fraction ($< \epsilon$) of the time, we're allowed to say we don't know what's going to happen. The rest of the time, we go back and ask our binary questions, compare our answers, and make our guess, and now we need to get the right answer.

That is, we don't have to get it right all the time, but we need to decide *at the end of the unary game* whether or not we're going to get it right, before we get to ask any binary questions.

If we can win this game, we can say the 3-graph is *admits linear error*.

The name "admits linear error" comes from this same idea in a slightly different context, so it's a little hard to justify.

The name “admits linear error” comes from this same idea in a slightly different context, so it’s a little hard to justify.

But for our purposes, it’s important to distinguish *unary* error, which comes from *one player’s* unary information, and *linear* error, which lets us use *everyone’s* unary information.

This new notion seems complicated, but it turns out to be easier to characterize than some of the others.

This new notion seems complicated, but it turns out to be easier to characterize than some of the others.

Remember the example $\mathcal{R}^3_{fun}$: we'll take X and Y to be the set of numbers up to n, and then take Z to be all the functions from X to Y. Then we say that x, y, z is an edge if $y < z(x)$.

This new notion seems complicated, but it turns out to be easier to characterize than some of the others.

Remember the example $\mathcal{R}^3_{fun}$: we'll take X and Y to be the set of numbers up to n, and then take Z to be all the functions from X to Y. Then we say that x, y, z is an edge if $y < z(x)$.

Theorem (Terry–Wolf)

$\mathcal{R}^3$ admits linear error if and only if there is some n so that $\mathcal{R}^3$ does not contain the $\mathcal{R}^3_{fun}$ example of size n.

Putting these together, these properties look like this: