An exponential improvement for Ramsey numbers

Talk by Julian Sahasrabudhe
Notes by Sanjana Das
March 1, 2024

§1 Introduction

Yesterday we saw a proof of the Erdős–Szekeres bound $R(k) \leq 4^k$. For the exponential improvement of $R(k) \leq (4-c)^k$, it actually makes sense not to directly think about the diagonal, but rather to think about the slightly *off-diagonal* case first. That's what we'll mainly focus on in today's talk — we'll consider $R(\ell, k)$ when $\ell = ck$ for a small (but fixed) constant c. In particular, we'll define the quantity

$$\gamma = \frac{\ell}{\ell + k}$$

(if $\ell = k$ then this would be $\frac{1}{2}$); throughout the talk we'll assume that γ is a small constant. It turns out that this makes certain things much easier (in particular, certain computations will work immediately, while they'd be much harder in the $\gamma = \frac{1}{2}$ case). And this is of interest especially because the proof in the diagonal case goes *through* the off-diagonal case — they first get prove the off-diagonal case, and then push the diagonal case to the off-diagonal case.

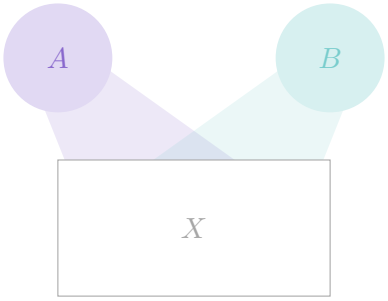
Yesterday, we briefly mentioned that the Erdős–Szekeres bound in the off-diagonal case gives $R(\ell, k) \leq \binom{k+\ell}{\ell}$. Our goal is to improve this to

$$R(\ell, k) \leq \binom{k+\ell}{\ell} e^{-\delta k},$$

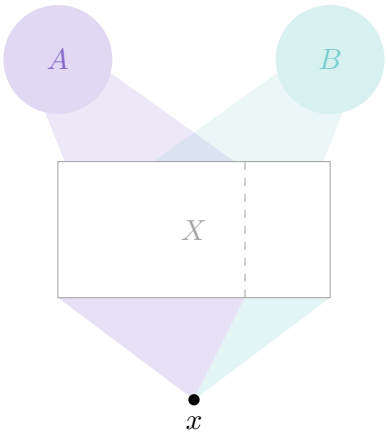
where δ is some constant depending on γ.

§1.1 The Erdős–Szekeres bound

First, let's remind ourselves of the proof of the Erdős–Szekeres bound (without the exponential improvement). We'll have a reservoir X of vertices in our graph (which starts out with size n), and as the algorithm goes along, we'll grow sets A and B. We'll maintain that A is a red clique and complete to X in red, and B is a blue clique complete to X in blue. (We'll use ℓ to refer to the blue color and k to the red color — so we're searching for a smaller blue clique or larger red clique.)



In a step of the algorithm, we take a vertex, pull it aside, and look at its red and blue neighbors.



Since we're working with the off-diagonal case, it makes sense to divide into cases a bit differently — instead of cutting at $\frac{1}{2}$, we'll cut at γ. So if x has at least $\gamma |X|$ blue neighbors in X, then we take a blue step (in which we move x to B and shrink X to just those blue neighbors); and if we have at least $(1-\gamma)|X|$ red neighbors, then we take a red step.

To analyze this, we started with n vertices. Every time we take a blue step we lose a factor of γ, and we take at most ℓ blue steps; every time we take a red step we lose a factor of $(1-\gamma)$, and we take at most k red steps. So for the algorithm to work, we need that

$$n\gamma^{\ell}(1-\gamma)^{k} \geq 1,$$

or equivalently

$$n \geq \gamma^{-\ell}(1-\gamma)^{-k} \approx \binom{k+\ell}{\ell}$$

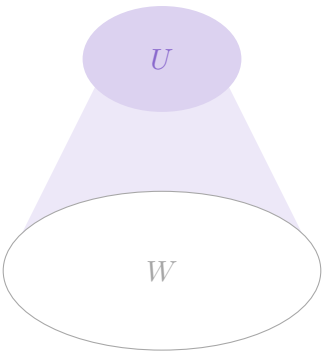
(we're hiding a polynomial term in this approximation, but since we're trying to get an exponential improvement, that doesn't really matter).

§1.2 Our setup

As mentioned yesterday, our goal is to take *lots* of red steps in some sense — more precisely, we'll create a 'book' structure that mimics this.

Definition 1.1. A *book* consists of two sets of vertices U and W such that U is a red clique, and all edges between U and W are red.

(There's no requirements on the edges inside W.)



We'll use t to denote the size of U. And we *want* to have $|W| \geq R(\ell, k-t)$. If we can get this, then we're done — we either get a blue ℓ-clique or a red $(k-t)$-clique in W, and in the latter case we can extend it to a red k-clique by adding in U.

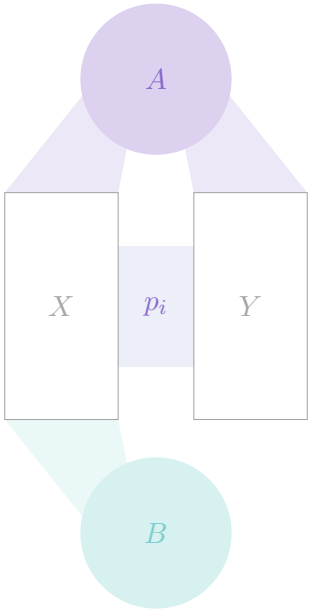
For our proof in the off-diagonal case, we'll simply use the Erdős–Szekeres bound on $R(\ell, k-t)$ — so we'll want to have

$$|W| \geq \binom{k+\ell-t}{\ell}.$$

§2 The book algorithm

We're now going to define how the algorithm runs. There'll be a few minor issues that we'll gloss over — we'll mention them and talk about why they're not major, but we'll mostly focus on the main issues (which will be the business of taking red steps and a tradeoff against a density increment, as mentioned yesterday).

We're looking to build a red book, and of course, we're also happy to build a blue clique. So we'll have a bunch of red vertices in one hand, joined in red to everything that remains (i.e., both X and Y); and we'll have a bunch of blue vertices in the other hand, asymmetrically joined to X in blue (but not necessarily Y). And the major departure from the Erdős–Szekeres proof is that we're also going to keep track of the red density between X and Y — we'll use p_i to denote this density on the ith step.



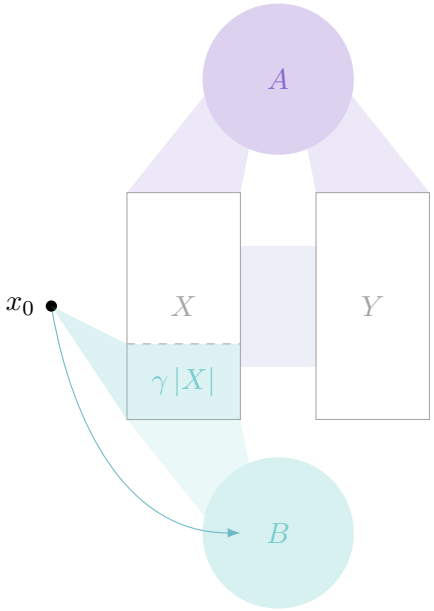
We'll think of X as our reservoir (so we're always choosing a vertex from X to move to A or B), and (A, Y) as the red book we're building up.

We'll initialize the algorithm by taking a random partition of the graph into X and Y, such that we have a healthy number of red edges between them (we won't focus on the initialization step right now; we'll come back to it later). Then as the algorithm goes along, there's several steps it'll perform, which we now describe.

§2.1 Blue steps

First, there's a 0th type of 'ghost step' which we'll come back to in a minute (it's needed to make the other steps work, but it's sort of a lower-order issue).

The first type of step is a *blue step* (this is sort of the simplest possible step). Suppose that we have a vertex $x_0 \in X$ with 'good' blue degree to X. First, what does 'good' mean? From the Erdős–Szekeres argument, we have a hint that it should probably be $\gamma |X|$. So in a blue step, if x_0 has blue degree at least $\gamma |X|$, then we'll move it to B and shrink X to just its blue neighbors (leaving Y as is).



This completely maintains our picture, except for potentially the (red) density between X and Y. But since we've only shrunk one side, the only reason we could have an issue here is if there's a bunch of vertices in X with low (red) degree to Y.

So to fix this, we get to the 0th step, the *degree regularization* step — where we just throw out vertices in X which have low red degree to Y. What do we mean by 'low'? We'd expect a vertex in X to have red degree roughly $p_i |Y|$ to Y. We'll throw out vertices whose degrees are even a tiny bit smaller than this — explicitly, we'll throw out vertices with red degree to Y less than

$$p_i |Y| (1 - \alpha_k k^{-1/8})$$

(where α_k is a moving target, but typically something like $1/k$).

With this out of the way, we can now take our blue step.

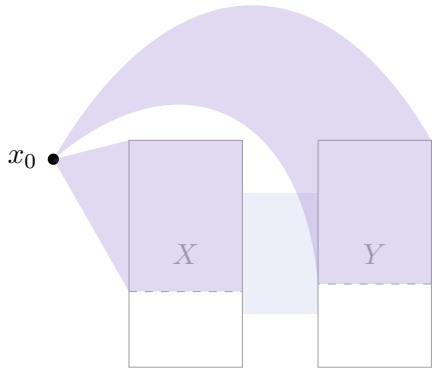
> **Algorithm 2.1** (Blue step) — If there exists $x_0 \in X$ such that $|N_B(x_0) \cap X| \geq \gamma |X|$, then:
>
> - Update $X \to X \cap N_B(x_0)$.
> - Update $B \to B \cup \{x_0\}$.

The point of degree regularization is that it ensures when we do this, p_i doesn't shrink too much.

§2.2 Red steps and density-boost steps

The next type of move is a *red step*. Suppose that on the 'left' side X, every vertex x_0 has blue neighborhood of size at most $\beta_i |X|$ in X, where $\beta_i < \gamma$ (otherwise we'd be able to take a blue step).

Thanks to the degree regularization step, we can also look at the red neighborhood of x on the 'right' side Y — i.e., $N_R(x_0) \cap Y$ — and we know this is of size roughly what we'd expect, i.e., $p_i |Y|$.



What we'd *like* to do is shrink X and Y down to the red neighborhoods of x_0 (both of which are healthy portions of their worlds). But the crux is, what if this drops the density between the new sets X and Y by a significant amount?

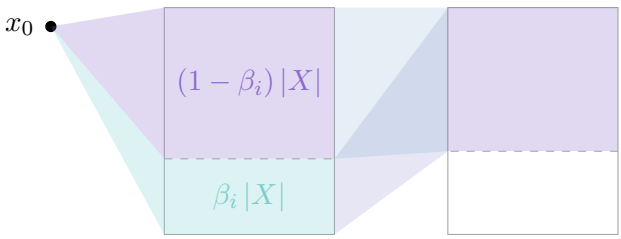
First, what's a 'significant amount'? We have to be careful about this, but heuristically it's not hard to guess what size of a drop would be a problem. First, we can expect our initial graph to have (red) density roughly $1-\gamma$. (If our density is significantly above this, then we can take a bunch of red steps in the Erdős–Szekeres algorithm, and if it's significantly below then we can take a bunch of blue steps; this would give us enough of a win. So we can assume that the graph has approximately the right density.)

Suppose that in a step, our density (between X and Y) drops by c/k. Is this too much to afford? The answer is yes — if we're losing a factor of $(1-c/k)$ at each step and we're running the algorithm for $\Theta(k)$ steps, then our density will have dropped by a *constant*. And that's catastrophic — because then we'll have an exponential loss in $|Y|$ (as we keep performing steps). So this is where we start having a problem. Conversely, if we can control the density better than this, then we should be okay (since the changes in density won't cause an exponential loss). So the problem occurs roughly at a drop of c/k.

> **Remark 2.2.** There are several places in which the talk references a small constant that I wrote down as c; so there's several c's floating through these notes that are not referring to the same constant.

So imagine that our enemy hands us precisely a drop of c/k in density; what should we do? We'll now write down *two* types of steps we could perform (there'll be a split in how we choose which one to do).

So the third type of step is a *density-boost step*. Where does the density boost come from in our picture? Imagine that the density between X and Y is $1-\gamma$. Then if the density between the red neighborhoods of x_0 in X and Y is $1-\gamma-\alpha$ (think of α as roughly c/k), then the density between the red neighborhood in Y and the *blue* neighborhood in X should be pretty large. (This isn't exactly true, but it's true for some choice of x.)



Specifically, we've lost α in the top density, so we should gain roughly

$$\frac{\alpha(1-\beta_i)}{\beta_i}$$

in the bottom density (the extra factor is because we're compensating with a much smaller portion of the world).

So we start to feel a bit that something good is happening, and we see why the off-diagonal is helpful here — if we were just on the diagonal, then β_i wouldn't be that small. But if γ is small then so is β_i, and so we actually pick up a bit of a win from the density-boost.

Then in a density-boost step, we're going to shrink X to the *blue* neighborhood of x_0 and Y to the *red* neighborhood, and pick up a gain in the red density; we'll also get to put x into B (giving us a small extra win).

Algorithm 2.3 (Density-boost step) — If the red density between $N_R(x_0) \cap X$ and $N_R(x_0) \cap Y$ is less than $p_i - \alpha_i$, then:

- Update $X \to X \cap N_B(x_0)$.
- Update $Y \to Y \cap N_R(x_0)$.
- Update $B \to B \cup \{x_0\}$.

As we'll see soon, we need to choose the value of α to depend on where we are in the algorithm (which is why we write α_i); but you can think of it as roughly c/k. (There's also some choice involved here with x_0, but we won't worry about that.)

Now we can describe the red steps too.

Algorithm 2.4 (Red step) — If the red density between $N_R(x_0) \cap X$ and $N_R(x_0) \cap Y$ is at least $p_i - \alpha_i$, then:

- Update $X \to X \cap N_R(x_0)$.
- Update $Y \to Y \cap N_R(x_0)$.
- Update $A \to A \cup \{x_0\}$.

§2.2.1 Choosing the cutoffs

First, what's stopping our enemy from just giving us a density-boost step every time? Often with density-boost arguments, you eventually just hit the maximum density (in which case you're done). But here, there's not enough time to hit the maximum density — we get a *little* boost in density, but if we think of α as c/k, then it's going to take $\Theta(k)$ time to hit the maximum, which is way too long. So we could imagine that our enemy just keeps pushing the density up and up, while really messing up everything else for you. (In other words, if you as the enemy keep handing me density-boost steps, I gain some density, but I can't do anything with it until I'm $\Theta(k)$ steps deep into the algorithm — there's no reason I should hit the maximum density, so I just keep getting handed density that doesn't seem to really be useful.)

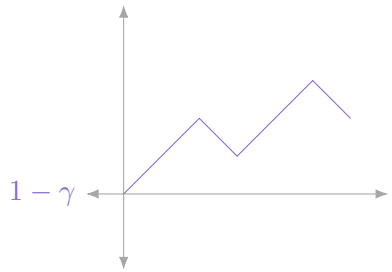
The idea to get around this issue is to choose the cutoffs α a bit dynamically (instead of having them be c/k for a fixed constant).

Remark 2.5. Regarding the initialization step, we can assume that the density in our original graph is close to $1-\gamma$ — if the red density is less than $1-\gamma-\eta$, where η is a small constant (small compared to our exponential win), then we have a great opportunity to just take a blue step — and we can keep on doing that until our density drifts back into the correct range. So we can assume the red density is at least something like $1-\gamma-\eta$ (the error from the $-\eta$ term will be small compared to our exponential win, so we'll basically ignore it). Then assuming this, we take a random partition into X and Y of equal size; then our red density is at least roughly $1-\gamma$, and we're off to the races.

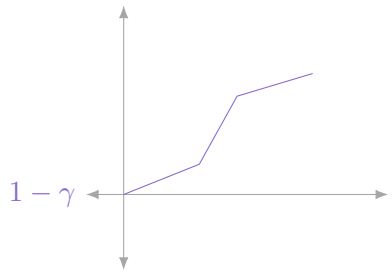
Remark 2.6. What's the significance of Y in the algorithm? We think of our blue clique as B (which we're building just from the reservoir X), and our red book as (A, Y). So we're trying really hard to keep Y as large as possible — it's sort of the most expensive real estate. (For X, we'll essentially just keep running the algorithm until it's dead.) And so the reason we care about the density between X and Y is so that when we add things to A, we don't shrink Y too much.

So now we'll tie up the loose end in the algorithm of how we choose α_i. The idea is that each step is kind of like a wager — I decide at step i how much I'm willing to lose (in density), and the enemy hands me a graph, and I make a decision based on that.

So we can imagine graphing the density vs. time. Imagine that our initial density is just $1 - \gamma$, and it'll bounce around as we move.



We want to avoid a scenario where the enemy keeps handing us density-boost steps. But if you hand me a bunch of density-boost steps, then my density has gone up by a lot; so I should also be more willing to *lose* density. In other words, if I'm much richer than I started off, then I'm happier to bet more — I'm happy to lose all this extra density that I gained, because I'm really only concerned with dropping below $1 - \gamma$.

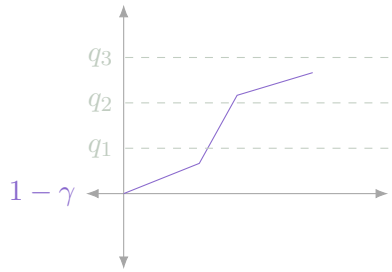


So we're going to develop some sort of 'levels' determining how much we're willing to bet at each stage. We define an exponentially growing sequence $q_1, q_2, \ldots$ so that the zone we're in determines how much we're willing to bet — we define

$$q_h = \frac{(1+\varepsilon)^h - 1}{k}$$

(the letter h is for 'height').

Remark 2.7. Our 'absolute scale' is something like $1/k$, corresponding to the heuristic from earlier; and we want something that grows exponentially as our height increases. The value of ε isn't super important, but it'll be something like $k^{-1/2}$.



These q_h's determine our regions, and the amount we bet in the hth region will be

$$\alpha_h = q_h - q_{h-1} = \frac{\varepsilon(1+\varepsilon)^{h-1}}{k}.$$

(So if our density is currently between q_{h-1} and q_h, then we bet α_h — these are what we fill in for the α_i's from earlier.)

The point of this definition is actually quite intuitive — what happens if our enemy keeps handing us density-boost steps? Then if we're currently in the hth level, we'll gain $q_h - q_{h-1}$ in density (really, we have an additional factor from β_i being small, but that only helps and isn't important here), and that'll jump us into the next region. And then if you hand me another density-boost step I again bet the length of the new region and jump into the next, and so on.

And $(1+\varepsilon)^h$ is growing exponentially in h, while our density has to be between 0 and 1. This means there are at most $\varepsilon^{-1}\log k = o(k)$ possible regions — so if the enemy keeps handing us density-boost steps, then we're going to be done (i.e., to hit maximum density) well before we're deep into the algorithm.

And the upshot of this is that we'll really be able to control the loss from the density-boost steps. (We'll come back to this picture soon.)

> **Remark 2.8.** Is there a way to see that even if the density always remains $1-\delta$ (i.e., the simplest case, where we don't ever need to take a density-boost step) then we win? In that situation, we're happy to just take a bunch of red steps, and so at the end of the day, we get a book with t vertices on the left. And on the right (in Y), we're losing a $1-\gamma$ factor every time we take a red step; so our book has t vertices on the left and $(1-\gamma)^t n$ on the right.
>
> (This is precisely the situation where we've created a perfect book — if we just had a random graph with density $1-\gamma$, then this is the book we'd create.)
>
> And if we can take $t \approx ck$, then this automatically gives you an exponential improvement (applying Erdős–Szekeres to bound the Ramsey number we want for Y). In some sense, this is the same thing as if I let you do Erdős–Szekeres but guarantee that you can take the first ck steps to all be red, and that's enough to give you an exponential improvement.
>
> Interestingly, in this situation we only need $t \approx \delta k$ (where δ is an arbitrarily small constant) to get an exponential win (something like $e^{-\delta\gamma k}$). But since we're going to be losing things in the density-boost steps (those steps also eat up Y), it's important that t is a *fixed* (not arbitrarily small) constant times k (e.g., $\frac{2}{3}k$). So we're going a bit further in building our book, but losing a bit more on the right side than you'd expect from the quasirandom situation — this is unusual compared to the usual book idea.

§3 Analysis

Now we'll analyze the algorithm. First let's track the evolution of the sizes of X and Y — there'll be some parameters we'll leave open for now, and we'll have to think about how to control them as time goes on.

§3.1 Size of X

We'll keep running the algorithm until X has basically been eaten up, and we want to understand what's happened at that point. First, what does X look like?

We'll say X has been eaten up if $|X| \leq 2^{o(k)}$. Over the course of the algorithm, we've taken some number of blue steps and lost a γ factor for each. We've taken at most ℓ blue steps, but we can actually say more than this because there's *two* ways to put a vertex into our blue set — both the genuine blue steps, and the density-boost steps.

Notation 3.1. We'll let s denote the number of density-boost steps, and $\mathcal{S}$ the set of indices i at which we've taken a density-boost step.

Remark 3.2. We use the letter s for density-boost steps because of 'booSt' — we can't use the letter b, because that's the first letter of 'blue.'

Then we've taken at most $\ell - s$ blue steps, losing a factor of $\gamma^{\ell - s}$.

And what about red steps? We'll let t be the number of red steps; we lose a $(1-\gamma)$ factor for each, so $(1-\gamma)^t$ in total.

The confusing thing is the density-boost steps. At each, we're shrinking to a β_i-proportion of our world; this gives a confusing product where we lose β_i at each step. This gives the following bound:

Lemma 3.3

At the end of the algorithm, we have

$$2^{o(k)} \geq |X| \geq \gamma^{\ell - s} \cdot (1-\gamma)^t \cdot \prod_{i \in \mathcal{S}} \beta_i \cdot n \cdot 2^{o(k)}.$$

(The $2^{o(k)}$ term on the right corresponds to lower-order fluctuations that we won't care about.)

There are two confusing things about this — we need a way of dealing with $\prod \beta_i$, and we also don't know what s is. (For that matter, we also don't know what t is.) We'll deal with both of these later.

§3.2 Size of Y

Now let's consider Y. When we take a red step, we're losing basically p_i, which we can think of as roughly $1-\gamma$; this means in total, the red steps lose roughly a factor of $(1-\gamma)^t$. (We haven't yet argued why the density doesn't drop much below this, but we'll explain this soon.)

And the other time we lose in Y, which is more painful, is when we take a density-boost step; here we again lose a factor of p_i. This gives us the following bound.

Lemma 3.4

At the end of the algorithm, we have

$$|Y| \geq \prod_{i \text{ red}} p_i \cdot \prod_{i \in \mathcal{S}} p_i \cdot n \cdot 2^{o(k)} \geq 2^{o(k)} (1-\gamma)^{t+s} n.$$

(The $2^{o(k)}$ again corresponds to lower-order terms that we ignore.)

In the perfect situation (e.g., a pseudorandom graph), we'd take *only* red steps (and not density-boost steps). But here we have an extra pain from s, so we need to control s.

First, let's explain why the second inequality in the lemma is true (i.e., why we can replace p_i with $1-\gamma$).

Lemma 3.5
For all i, we have $p_i \geq (1-\gamma) - o(1)$.

(Here the term of $(1-\gamma)$ corresponds to our initial value of p — which we've argued can roughly be assumed to be $1-\gamma$. The $o(1)$ term basically corresponds to ε.)

Proof. The proof is a bit confusing because we have different kinds of steps, and we need to ensure that they don't combine in some weird adversarial way. But essentially the only way we lose density is when we take red steps — the density-boost steps only push the density up. There's also the degree-regularization step, which also pushes density up. We do lose a little bit from the blue steps, but we'll forget about that.

Remark 3.6. There's a little bit of a cheat we've been doing here — we've basically been assuming that we never encounter a regularization step. In reality, we'll be doing some regularization steps in between (they're not a problem for this part of the analysis, because our degree only goes up).

So we're doing a slightly simplified analysis here where we assume everything's regular all the time (and ignore those steps). But this isn't too much of a cheat. There *is* a slight technicality where if we do degree-regularization and then one blue step, we lose a bit too much. But the way we fix this is that when we take blue steps, instead of taking just one, we take a whole bunch of blue steps (roughly $\sqrt{k}$) at once. Then the enemy can hand us at most $\sqrt{k}$ such steps over the whole course of the algorithm; and so this loss is a lower-order term.

So we can zoom around in lots of confusing ways, but from the perspective of degrees, basically the worst thing that can happen is if we take a bunch of red steps and our density keeps going down. Our α_h bottom out at $h=1$, so the bottom rung of our ladder has an α-value of ε/k. Then since we have k steps and we lose at most ε/k at each one of them, our total loss in density is at most ε; and since $\varepsilon \to 0$ as $k \to \infty$, this loss in density is $o(1)$. □

Remark 3.7. This is a bit like playing poker where you just fold every time. (And density-boost steps are like when you win some things, so you start wagering stuff.)

This justifies the business where we can just replace the p_i's with $1-\gamma$ (in the statement of the lemma) — each p_i is $1-\gamma-o(1)$, and we're raising this to a power of around k, so the $o(1)$ terms contribute a sub-exponential factor.

§3.3 Controlling β_i and s

Now the next step shows how to control both s and $\prod_{i\in\mathcal{S}} \beta_i$. *A priori* our enemy could really devastate $|X|$ with the β_i's, but this somehow plays against a win in the density.

Lemma 3.8
We have $\sum_{i\in\mathcal{S}} \frac{1-\beta_i}{\beta_i} \leq t + o(k)$.

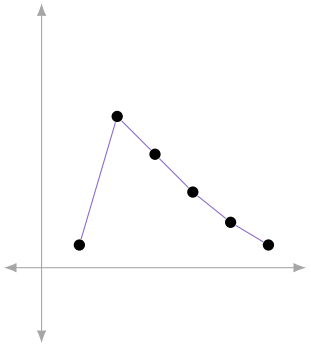
Before we get into the proof, we can see that this somehow controls how small the β_i's can be in terms of our favorite thing, which is taking red steps.

Proof. Writing down a full proof is a bit annoying because of the different types of steps pushing us around in the picture, but here's the basic idea. Let's go back to our graph of density vs. time — we start out with density $1-\gamma$, and the density bounces around as the algorithm goes along.

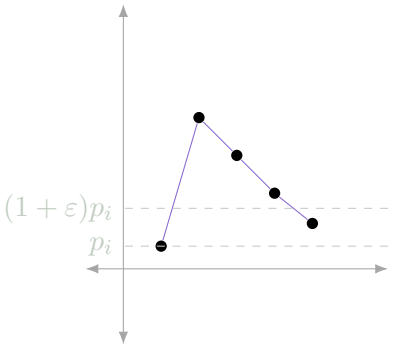
Suppose that we're handed a density-boost step. Then our density doen't go up just by α_h, but actually by

$$\alpha_h \cdot \frac{1-\beta_i}{\beta_i}.$$

And how many red steps does it take us to get back to where we originally were? Assuming that everything is happening in the same level (this makes sense because ε is small, so things are locally basically linear, and it's fine to assume this), at every red step we're losing α_h. So then it takes us at least $\frac{1-\beta_i}{\beta_i}$ red steps to get back down.



This isn't quite a proof (though this is the right ratio), because what if we never actually come back down? So let's ask a slightly different question — how much time does it take us to come down up to a factor of $(1+\varepsilon)$ of where we started (with the same ε as before)? Again, we're jumping up by $\alpha_h \cdot \frac{1-\beta_i}{\beta_i}$ and move down by α_h, so it still takes roughly $\frac{1-\beta_i}{\beta_i}$ steps to get back down.



But then you can say, what if we never come back to this region either? This means that forever, we'll be living above the line at $1+\varepsilon$ — in other words, we've sort of moved up a level q_h. But we can't move up a level very much, because there's $o(k)$ different levels. So this just contributes to the $o(k)$ term; and that's basically the proof. (The actual proof is more complicated because of the different kinds of steps, but this is the idea.) □

From this, we get a couple of really nice things. First recall that $\beta_i \leq \gamma$. So this automatically gives us some control on the *number* of density-boost steps.

Corollary 3.9
We have $(\frac{1-\gamma}{\gamma})s \leq \sum_{i\in\mathcal{S}} \frac{1-\beta_i}{\beta_i} \leq t + o(k)$.

Now we can use this to get a better handle on $|X|$.

§3.4 Putting things together

All the ideas have been presented now, and the rest is just a matter of putting things together; but we'll go through this to give a feel of what's pushing and pulling in the proof.

In our bound for $|X|$, we have a product of β_i's that we'd like to control. We'll use AM-GM — we have

$$\prod_{i\in\mathcal{S}} \frac{1}{\beta_i} \leq \left(\frac{1}{s}\sum_{i\in\mathcal{S}} \frac{1}{\beta_i}\right)^s = \left(\frac{1}{\beta}\right)^s$$

(where we define $\beta = \frac{1}{s}\sum_{i\in\mathcal{S}} \frac{1}{\beta_i}$ for convenience — so β is a sort of average of the β_i's'). This is an upper bound on the product of reciprocals, so it's a lower bound on the normal product, and we get

$$|X| \geq \gamma^{\ell-s}(1-\gamma)^t\beta^s \cdot n \cdot 2^{o(k)} = \gamma^\ell(1-\gamma)^t\left(\frac{\beta}{\gamma}\right)^s \cdot n \cdot 2^{o(k)}.$$

And we know $|X| \leq 2^{o(k)}$; we want to use this to conclude that we've then taken quite a few red steps.

We *want* to get an exponential win, so we want to plug in $n = \binom{k+\ell}{\ell}e^{-\delta k}$ (for some small $\delta > 0$). We have $\binom{k+\ell}{\ell} \approx \gamma^{-\ell}(1-\gamma)^{-k}$ (up to sub-exponential factors), and plugging this in, stuff cancels and we get that

$$2^{o(k)} \geq (1-\gamma)^{-k+t}\left(\frac{\beta}{\gamma}\right)^s e^{-\delta k}.$$

We want to turn this into a bound on t. Since γ is small, we have $1-\gamma \approx e^{-\gamma}$, which gives

$$2^{o(k)} \geq \exp\left(\gamma(k-t) - s\log\frac{\gamma}{\beta} - \delta k\right).$$

We're not going to do all the details, but to deal with the middle term, we use our upper bound on s from above; this allows us to replace the term with a bound something like

$$s\log\frac{\gamma}{\beta} \geq \frac{\beta}{1-\gamma}t\log\frac{\gamma}{\beta}.$$

We want to get rid of β completely; to do so, we can optimize this quantity in terms of β, and find that the optimum (for our enemy) is at $\beta = \gamma/e$. Plugging this into our middle term, we get that

$$2^{o(k)} \geq \exp\left(\gamma(k-t) - \frac{\gamma t}{e(1-\gamma)} - \delta k\right).$$

And the upshot is that in order for this to be $o(k)$, we need t to be able to compte with the k in the front; and so as long as γ is small, we get $t \geq k/2$ or something similar. So the point of this is that we're really taking quite a few red steps.

Remark 3.10. Think of δ as proportional to γ (i.e., γ times a sufficiently small constant).

We'll now set up the final check; we won't do each step, but it's informative to see how things come together.

We've taken t steps and created a book with t vertices on the one side, and Y on the other; and we've lower-bounded $|Y|$. Now we need to certify that $|Y| \geq R(\ell, k-t)$. So it's enough to show that $2^{o(k)}(1-\gamma)^{t+s}n$ (the quantity in our lower bound) is at least the bound coming from Erdős–Szekeres on $R(\ell, k-t)$ — we need to check that

$$2^{o(k)}(1-\gamma)^{t+s}n \geq \binom{k+\ell-t}{\ell} \geq R(\ell, k-t).$$

And if $\delta = c\gamma$ for a small constant, then this does work; and we get the exponential improvement.

Remark 3.11. Is this obvious, or do we just need to do the computation? It's mechanical in a sense, but not obvious. If you ignore s (i.e., there's no density-boost steps), then one way to think of this is that the γ in the new situation (the new Ramsey number being computed) is different from the γ in the old situation. Explicitly, we can rewrite

$$\binom{k+\ell-t}{\ell} \approx \binom{k+\ell}{\ell}(1-\gamma)^t \exp\left(-\frac{\gamma t^2}{2k}\right).$$

This means we get a bit of an extra win if t is linear in k. So putting back in the density-boost steps, we just need to check that

$$(1-\gamma)^s e^{-\delta k} \geq \exp\left(-\frac{\gamma t^2}{2k}\right)$$

(i.e., our loss from the density-boost steps is compensated from by our gain here). And this comes from the same inequality on s. Here it matters that t is a genuine constant (e.g., $1/2$) times k (though it doesn't matter exactly what that constant is, e.g., whether it's $1/2$ or $1/3$).

§4 Comments on the diagonal case

Finally, we'll finish by saying a few words about how to get to the diagonal case.

§4.1 Approach from yesterday's conjecture

We can try to modify the algorithm a bit to directly improve $R(k)$. As a simplified version, if the blue neighbors of our vertex x are at least a $(\frac{1}{4}+c)$-fraction of X, then we'll take a blue step. This will be enough to get an exponential win if we only want to take $t = \delta k$ red steps (where c depends on δ).

What does a red step look like? Our vertex x has a *huge* red neighborhood — a $(\frac{3}{4}-c)$-fraction — and we want to again perform a red step where we shrink the two sides, and maintain density $\frac{1}{2}$ between them.

If we *can't* choose a vertex where we get good control, then all the red neighborhoods of the potential vertices are sort of repulsive on a scale of $1-c/k$. But this has to happen for nearly $3/4$ of all the vertices in X (we're only taking a *few* red steps here, so we don't mind shrinking X substantially on those steps; so the only issue is if nearly all $3/4-c$ are repulsing each other).

And that's where the conjecture comes from — it seems like the only way this can happen is if there exists some 'clump' in the graph. We'll end by stating a graph-theoretic conjecture we don't know how to prove, which would imply that this algorithm just works for the diagonal case.

Definition 4.1. Given a bipartite graph on sets X and Y where each $x \in X$ has neighborhood roughly a p-fraction of Y, we say two vertices $x, y \in X$ are *ε-shy* if their neighborhoods have intersection at most a $p^2(1-\varepsilon)$-fraction of Y.

(We really care about the scale where $\varepsilon \approx c/k$.)

> **Conjecture 4.2 —** Suppose that G is a bipartite graph for which for all $x \in X$, all but a $(\frac{1}{4}+c)$-fraction of $y \in X$ are ε-shy of x. Then for any constant $L > 1$, there exists t and some $X' \subseteq X$ and $Y' \subseteq Y$ of sizes $e^{-ct}|X|$ and $e^{-ct}|Y|$ such that the density between them is at least $\frac{1}{2}(1 + \varepsilon t \cdot L)$.

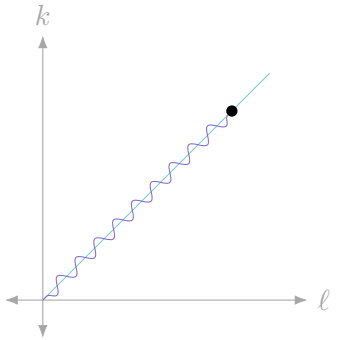
(You can get a gain at εt without too much trouble, but we want to beat that by a large constant factor.)

If we could get something like this, then we could plug it in and get the diagonal story to work — our blue steps are a bit bigger than $\frac{1}{4}$, and on red steps we take a vertex with large red degree, throw out lots of vertices that don't correlate with it and pass X to a small correlating subset, and pass Y to its neighborhood on the right. If we can't do this, that means almost all the vertices in X are anti-correlated (i.e., ε-shy, with $\varepsilon \approx c/k$). And if the conjecture holds, then we can get a big density boost by passing to X' and Y'.

§4.2 The actual approach

What do we change to get the diagonal case? We run the same algorithm, but instead of running it with bias γ vs. $1-\gamma$ (which would be a 50-50 split), we use a lopsided choice — we choose our breaking point to be 2/5 vs. 3/5. The idea is that this sort of biases the algorithm to run to the off-diagonal — and we still get a book with this funny choice of parameter, with t pretty large (something like $t \approx 2k/3$). If we just applied the Erdős–Szekeres bound to this remaining bit, then it wouldn't be enough. But we've taken so many red steps in this initial run that we're actually pretty far off the diagonal — we've gotten to something like $R(k/3, k)$. And in that situation, our bounds from the off-diagonal case are much better. And so the win we get from our exponential improvement compensates from the loss we used to push ourselves off the diagonal. (There's a bit of a technical challenge where we just know of a *region* of pairs (s,t) that are possible outcomes, so we need to check for this whole region that the corresponding bound is good enough.)

As a picture to keep in mind, imagine we plot k vs. ℓ (so the diagonal Ramsey numbers correspond to the diagonal). We're starting on this diagonal; a bad situation for Erdős–Szekeres is when we're forced to take a blue step, then a red step, then a blue step, and so on, so we're wiggling down the diagonal.



What happens for us is that we use our algorithm to take a big jump *off* the diagonal, bringing us to $(k-t, k)$. And then we run our algorithm again for the *off-diagonal* case to take a big step left, and the win we pick up from here is enough to just finish off with Erdős–Szekeres.

> **Remark 4.3.** There's another interesting twist that happens if you want Ramsey numbers very *close* to the diagonal, but not exactly on it. This is because on the diagonal there's symmetry between the two colors, so we can e.g. just say that one has density at least $\frac{1}{2}$ and take that to be red. But when you break symmetry, there's a problem — you have to consider whether the blue edges are a bit heavier or lighter than they should be. So here we have a different proof where we use another method of creating a *blue* book that lets us jump to the diagonal; and then we jump off the diagonal in this way again.