Improved Replica Bounds for the Independence Ratio of Random Regular Graphs

Viktor Harangi[1]

Abstract

Studying independent sets of maximum size is equivalent to considering the hard-core model with the fugacity parameter λ tending to infinity. Finding the independence ratio of random d-regular graphs for some fixed degree d has received much attention both in random graph theory and in statistical physics. For $d \geq 20$ the problem is conjectured to exhibit 1-step replica symmetry breaking (1-RSB). The corresponding 1-RSB formula for the independence ratio was confirmed for (very) large d in a breakthrough paper by Ding, Sly, and Sun. Furthermore, the so-called interpolation method shows that this 1-RSB formula is an upper bound for each $d \geq 3$. For $d \leq 19$ this bound is not tight and full-RSB is expected. In this work we use numerical optimization to find good substituting parameters for discrete r-RSB formulas ($r = 2, 3, 4, 5$) to obtain improved rigorous upper bounds for the independence ratio for each degree $3 \leq d \leq 19$. As r grows, these formulas get increasingly complicated and it becomes challenging to compute their numerical values efficiently. Also, the functions to minimize have a large number of local minima, making global optimization a difficult task.

1 Introduction

This paper is concerned with the independence ratio of random regular graphs. A graph is said to be *regular* if each vertex has the same degree. For a fixed degree d, let $\mathbb{G}(N, d)$ be a uniform random d-regular graph on N vertices. Note that $\mathbb{G}(N, d)$ has a "trivial local structure" in the sense that with high probability almost all vertices have the same local

✉ Viktor Harangi
    harangi@renyi.hu

[1] Alfréd Rényi Institute of Mathematics, Budapest, Hungary

neighborhood: $\mathbb{G}(N, d)$ almost surely *converges locally* to the d-regular tree T_d as $N \to \infty$. In statistical physics T_d is also known as the *Bethe lattice*. In fact, Mézard and Parisi used the expression Bethe lattice for referring to $\mathbb{G}(N, d)$ (see [21] for example), and proposed to study various models on these random graphs.

In a graph, an *independent set* is a set of vertices, no two of which are adjacent, that is, the induced subgraph has no edges. The *independence ratio* of a graph is the size of its largest independent set normalized by the number of vertices. For any fixed degree $d \geq 3$, the independence ratio of $\mathbb{G}(N, d)$ is known to converge to some constant α_d^* as $N \to \infty$ [4]. Determining α_d^* is a major challenge of the area. The cavity method, a non-rigorous statistical physics tool, led to a 1-step replica symmetry breaking (1-RSB) formula for α_d^*. The authors of [5] also argued that the formula may be exact for $d \geq 20$, which is widely believed to be indeed the case. Later this 1-RSB formula was confirmed to be exact for (very) large d in the seminal paper of Ding et al. [10].

Lelarge and Oulamara [19] used the *interpolation method* to rigorously establish[1] the 1-RSB formula as an upper bound for every $d \geq 3$. This approach also provides r-step RSB bounds for any $r \geq 2$. The problem is that these formulas get increasingly complicated and fully solving the corresponding optimization problems seems to be out of reach. Can we, at least, get an estimate or a bound?

The parameters of the r-RSB bound includes a *functional order parameter* which can be thought of as a measure.[2] The optimal measure satisfies a certain self-consistency equation. We cannot hope for an exact solution so the natural instinct is to try to find an approximate solution. In the physics literature an iterative randomized algorithm called *population dynamics* is often used to find an approximate solution in the 1-RSB (and occasionally in the 2-RSB) setting for various models. This sounds like a promising approach but we came to the surprising conclusion that for the hard-core model it may be a better strategy to forget about the equation altogether and search among "simple" measures. It seems to be possible to get very close to the global optimum using atomic measures with a moderate number of atoms. Furthermore, when we only have a few atoms, we can tune their weights and locations to a great precision, and this seems to outweigh the advantage of having a more "delicate" measure (but being unable to tune it to the same precision).

Moreover, using a small number of atoms means that we can compute the value exactly and the interpolation method ensures that what we get is always a rigorous upper bound. In contrast, population dynamics only gives an estimate for the value of the bound because for large populations one simply cannot compute the corresponding bound precisely and has to settle for an estimate based on a sample.

Therefore, our approach is that we try to find local minima of the discrete version (corresponding to atomic measures) using a computer. Even this is a formidable challenge as we will see. Table 1 shows the best bounds we found via numerical optimization.

These may seem to be small improvements but we actually expect the true values to be fairly close to our new bounds. In particular, for $d = 3$ it is reasonable to conjecture that the bound is sharp up to at least five decimal digits, that is, $\alpha_3^* = 0.45078....$

We also have improvements for $9 \leq d \leq 19$. However, as the degree gets closer to the threshold $d \geq 20$ (above which 1-step replica symmetry breaking is believed to be the truth),

---

[1] Their work builds on [12, 13, 23]. In fact, [5] already says that the 1-RSB formula is known to be an upper bound due to the Franz–Leone–Panchenko–Talagrand theorem.

[2] More precisely, as a measure that is supported on measures that are supported on measures that ... (iterated r times).

Table 1 Upper bounds for the asymptotic independence ratio α_d^*, $3 \leq d \leq 8$

	$d = 3$	$d = 4$	$d = 5$	$d = 6$	$d = 7$	$d = 8$
1-RSB	0.45085966	0.41119457	0.37926817	0.35298455	0.33088436	0.31197257
r-RSB	0.45078521	0.41109414	0.37917031	0.35289949	0.33081722	0.31192223
	(5-RSB)	(4-RSB)	(4-RSB)	(4-RSB)	(4-RSB)	(3-RSB)

We indicated which r-RSB formula yielded our bound. Note that this is not our estimate for the exact r-RSB value, but the best upper bound we found

the 1-RSB bound gets sharper and our improvement gets smaller. For more details, see the tables in the Appendix.

1.1 Upper Bound Formulas

In Sect. 2 we will explain the RSB bounds in detail. Here we only display a few formulas in order to give the reader an idea of the optimization tasks we are faced with.

For comparison, we start with the replica symmetric (RS) bound: for any $\lambda > 0$ and any $x \in [0, 1]$ we have

$$\alpha_d^* \log \lambda \leq \log \left(1 + \lambda(1 - x)^d\right) - \frac{d}{2} \log(1 - x^2). \tag{1}$$

Here the fugacity parameter λ is the "reward" for including a vertex in the independent set, while x can be thought of as the probability that a *cavity*[3] vertex is included. Then the right-hand side expresses the change in the *free energy* when adding a star (i.e., connecting a new vertex to d cavities) versus adding $d/2$ edges between cavities. Choosing λ and x optimally leads to the exact same formula as the Bollobás bound from 1981 [6], which was based on a first moment calculation for the number of independent sets of a given size. Actually, this relatively simple bound is already asymptotically tight: $(2 + o_d(1)) \frac{\log d}{d}$, where the asymptotic lower bound is due to Frieze and Łuczak [11].

The 1-RSB bound says that for any $\lambda_0 > 1$ and any $q \in [0, 1]$:

$$\alpha_d^* \log(\lambda_0) \leq \log \left(1 + (\lambda_0 - 1)(1 - q)^d\right) - \frac{d}{2} \log \left(1 - (1 - 1/\lambda_0)q^2\right). \tag{2}$$

Choosing λ_0 and q optimally leads to an (implicit) formula for α_d^*. As we mentioned, this 1-RSB bound is conjectured to be sharp for any $d \geq 20$ and known to be sharp for sufficiently large d.

Heavy notation would be needed to describe the r-step RSB bounds in general. In order to keep the introduction concise, we only give (a discretized version of) the formula for the case $r = 2$: for any $\lambda_0 > 1$, $0 < m < 1$, and any $p_1, \ldots, p_n, q_1, \ldots, q_n \in [0, 1]$ with $p_1 + \cdots + p_n = 1$ we have

$$\alpha_d^* m \log(\lambda_0) \leq \log \sum_{i_1=1}^{n} \cdots \sum_{i_d=1}^{n} \left(\prod_{\ell=1}^{d} p_{i_\ell}\right) \left(1 + (\lambda_0 - 1) \prod_{\ell=1}^{d} (1 - q_{i_\ell})\right)^m$$

$$- \frac{d}{2} \log \sum_{i_1=1}^{n} \sum_{i_2=1}^{n} p_{i_1} p_{i_2} \left(1 - (1 - 1/\lambda_0) q_{i_1} q_{i_2}\right)^m. \tag{3}$$

---

[3] Cavities are vertices of degree $d - 1$ that are created by deleting a relatively small number of edges from the graph. The point is that this way the symmetry is broken while the independence ratio is essentially unchanged.

Table 2 The degree 3 case: our best r-RSB bounds for α_3^* for $r = 2, 3, 4, 5$ compared to previous upper bounds

RS/first moment	0.45906	Bollobás [6]
McKay bound	0.45537	McKay [20]
1-RSB	0.45085966	Lelarge–Oulamara [19]
2-RSB	0.45078994	
3-RSB	0.45078602	
4-RSB	0.45078535	
5-RSB	0.45078521	

Table 3 Lower bounds on α_3^*

0.4328	Hoppen [16]
0.4352	Kardoš–Král–Volec [18]
0.4361	Csóka–Gerencsér–Harangi–Virág [7]
0.4380†	
0.4375†	Hoppen–Wormald [17]
0.4453†	Csóka [9]

Note that the bounds marked with † cannot be considered fully rigorous as they require some kind of computer simulation or estimation

The number of parameters for general r is roughly $2n_1 \ldots n_{r-1}$, where n_k denotes the number of atoms used at the different layers, so the dimension of the parameter space grows exponentially in r, see Sect. 2.3 for details.

1.2 The Case of Degree 3

One can plug any concrete choice of parameter values into (3) to get a bound for the independence ratio. To demonstrate the strength of (3) even for small n, we include here an example for a 2-RSB bound for $d = 3$, $n = 4$: the values

$$\lambda_0 = 19.3 \qquad p_1 = 0.2493 \qquad p_2 = 0.2778 \qquad p_3 = 0.2880 \qquad p_4 = 0.1849$$
$$m = 0.557 \qquad q_1 = 0.1184 \qquad q_2 = 0.5947 \qquad q_3 = 0.8876 \qquad q_4 = 0.9827$$

give a bound $\alpha_3^* < 0.450789952 < 0.45079$ that already comfortably beats the currently best bound (≈ 0.45086). Table 2 shows our best bounds for $d = 3$.

As for lower bounds for small d, the best results have been achieved by so-called *local algorithms*. Table 3 lists a few selected works and the obtained bounds for α_3^*.

Note that a beautiful result of Rahman and Virág [25], building on a work of Gamarnik and Sudan [14], says that asymptotically (as $d \to \infty$) local algorithms can only produce independent sets of half the maximum size (over random regular graphs). For small d, however, the independence ratio produced by local algorithms may be the same as (or very close to) α_d^*.

1.3 Optimization

We wrote Python/SAGE codes to perform the numerical optimization for the replica bounds.

- The first task was to efficiently compute the r-RSB formulas and their derivatives w.r.t. the parameters.

- Then we used standard algorithms to perform local optimization starting from random points. As the parameter space grows, more attempts are required to find an appropriate starting point leading to a good local optimum.
- Eventually we start to encounter a *rugged landscape* with a huge number of local minima, where we cannot expect to get close to the global optimum even after trying a large number of starting points. In order to overcome this obstacle, for $d = 3$ we used a technique called *basin hopping*. In each step, the algorithm randomly visits a "nearby" local minimum, favoring steps to smaller values. This approach led to the discovery of our best bounds for $d = 3$.
- The smaller the degree d, the deeper we could go in the replica hierarchy (i.e., use larger r). We could perform the 3-RSB optimization for $d \leq 10$, the 4-RSB optimization for $d \leq 6$, and the 5-RSB optimization for $d = 3$.

See Sect. 3.2 for further details about the implementation.

Although the bounds are hard to find, they are easy to check: one simply needs to plug the specific parameter values into the given formulas. We created a website with interactive SAGE codes where the interested reader may check the claimed bounds and even run simple optimizations: https://www.renyi.hu/~harangi/rsb.htm. Our codes can be found in the public GitHub repository https://github.com/harangi/rsb.

1.4 2-RSB in the Literature

As far as we know, there was only one previous attempt to get an estimate for the 2-RSB formula (only for $d = 3$). In [5] it reads that "the 2-RSB calculation is [...] somewhat involved and was done in [24] [and obtained the value] 0.45076(7)". Rivoire's thesis [24] indeed reports briefly of a 2-RSB calculation. Note that, since he considers the equivalent vertex-cover problem (concerning the complements of independent sets), we need to subtract his value from 1 to get our value. On page 113 he writes that using population dynamics he obtained the following estimate: 0.54924 ± 0.00007. For our problem this means $0.45076 \pm 0.00007 =$ [0.45069, 0.45083]. The value 0.45076(7) in [5] may have come from mistakenly using an error ± 0.000007 instead of ± 0.00007 when citing Rivoire's work. The thesis only provides a short description of how this estimate was obtained. The author refers to it as "unfactored" 1-RSB and it seems to be the same as what we call a non-standard 1-RSB in our remarks after Theorem 2.2. If that is indeed the case, then our findings suggest that its true value should actually be around 0.45081.

1.5 Outline of the Paper

In Sect. 2 we present the general replica bounds and their discrete versions that we need to optimize. Section 3 contains details about the numerical optimization. In Sect. 4 we revisit the $r = 1$ case and investigate more sophisticated choices for the functional order parameter. The Appendix contains a table listing our best bounds for different values of d and r (Sect. 5) and an overview of the interpolation method for the hard-core model over random regular graphs (Sect. 6).

2 Replica Formulas

Originally the cavity method and belief propagation were non-rigorous techniques in statistical physics to predict the free energy of various models. They inspired a large body of rigorous work, and over the years several predictions were confirmed. In particuler, the so-called *interpolation method* has been used with great success to establish rigorous upper bounds on the free energy.

In the context of the hard-core model over random d-regular graphs, the interpolation method was carried out by Lelarge and Oulamara in [19], building on the pioneering works [12, 13, 23]. First we present the general r-step RSB bound obtained this way.

2.1 The General Replica Bound

For a topological space Ω let $\mathcal{P}(\Omega)$ denote the space of Borel probability measures on Ω equipped with the weak topology. We set $\mathcal{P}^1 := \mathcal{P}([0, 1])$ and then recursively $\mathcal{P}^{k+1} := \mathcal{P}(\mathcal{P}^k)$ for $k \geq 1$. The general bound will have the following parameters:

- $\lambda > 1$;
- $0 < m_1, \ldots, m_r < 1$ corresponding to the so-called Parisi parameters;
- a measure $\eta^{(r)} \in \mathcal{P}^r$.

Definition 2.1 Given a fixed $\eta^{(r)} \in \mathcal{P}^r$, we choose (recursively for $k = r - 1, r - 2, \ldots, 1$) a random $\eta^{(k)} \in \mathcal{P}^k$ with distribution $\eta^{(k+1)}$. Finally, given $\eta^{(1)}$ we choose a random $x \in [0, 1]$ with distribution $\eta^{(1)}$. In fact, we will need d independent copies of this random sequence, indexed by $\ell \in \{1, \ldots, d\}$. Schematically:

$$\eta^{(r)} \rightarrow \eta^{(r-1)}_\ell \rightarrow \cdots \rightarrow \eta^{(1)}_\ell \rightarrow x_\ell \quad (\ell = 1, \ldots, d).$$

For $1 \leq k \leq r$ we define $\mathcal{F}_k$ as the σ-algebra generated by $\eta^{(r-1)}_\ell, \ldots, \eta^{(k)}_\ell, \ell = 1, \ldots, d$, and by $\mathbb{E}_k$ we denote the conditional expectation w.r.t. $\mathcal{F}_k$. Note that $\mathcal{F}_r$ is the trivial σ-algebra and hence $\mathbb{E}_r$ is simply $\mathbb{E}$.

Given a random variable V (depending on the variables $\eta^{(k)}_\ell, x_\ell$), let us perform the following procedure: raise it to power m_1, then apply $\mathbb{E}_1$, raise the result to power m_2, then apply $\mathbb{E}_2$, and so on. In formula, let $T_0 V := V$ and recursively for $k = 1, \ldots, r$ set

$$T_k V := \mathbb{E}_k (T_{k-1} V)^{m_k}.$$

In this scenario, applying $\mathbb{E}_k$ means that, given $\eta^{(k)}_\ell, \ell = 1, \ldots, d$, we take expectation in $\eta^{(k-1)}_\ell, \ell = 1, \ldots, d$ (or in x_ℓ if $k = 1$).

Now we are ready to state the r-RSB bound given by the interpolation method.

Theorem 2.2 *Let $r \geq 1$ be a positive integer and $\lambda, m_1, \ldots, m_r, \eta^{(r)}$ parameters as described above. Let $x_\ell, \ell = 1, \ldots, d$ denote the random variables obtained from $\eta^{(r)}$ via the procedure in Definition 2.1. Then we have the following upper bound for the asymptotic independence ratio α^*_d of random d-regular graphs:*

$$\alpha^*_d m_1 \cdots m_r \log \lambda \leq \log T_r \big(1 + \lambda(1 - x_1) \cdots (1 - x_d)\big) - \frac{d}{2} \log T_r (1 - x_1 x_2).$$

This was rigorously proved in [19]. They actually considered a more general setting incorporating a class of (random) models over a general class of random hypergraphs (with

given degree distributions). They used the hard-core model over d-regular graphs as their chief example, working out the specific formulas corresponding to their general RS and 1-RSB bounds. Theorem 2.2 follows from their general r-RSB bound [19, Theorem 3] exactly the same way as in the RS and 1-RSB case.

We should make a number of remarks at this point.

- Above we slightly deviated from the standard notation as the usual form of the Parisi parameters would be

$$0 < \hat{m}_1 < \cdots < \hat{m}_r < 1,$$

where $\hat{m}_k$ can be expressed in terms of our parameters m_k as follows:

$$\hat{m}_r = m_1; \quad \hat{m}_{r-1} = m_1 m_2; \quad \ldots; \quad \hat{m}_1 = m_1 m_2 \cdots m_r.$$

As a consequence, the indexing of $\mathcal{F}_k$, $\mathbb{E}_k$, T_k is in reverse order, and the definition of T_k simplifies a little because raising to power $1/\hat{m}_{r-k+2}$ and then immediately to $\hat{m}_{r-k+1}$ (as done, for example, in [23]) amounts to a single exponent $\hat{m}_{r-k+1}/\hat{m}_{r-k+2} = m_k$ in our setting.

- Also, generally there is an extra layer of randomness (starting from an $\eta^{(r+1)} \in \mathcal{P}^{r+1}$) resulting in another expectation outside the log. This random choice is meant to capture the local structure of the graph in a given direction. However, when the underlying graph is d-regular (meaning that essentially all vertices see the same graph structure locally), we do not need this layer of randomness (in principle). Therefore, in the d-regular case one normally chooses a trivial $\eta^{(r+1)} = \delta_{\eta^{(r)}}$. That is why we omitted $\eta^{(r+1)}$ and started with a deterministic $\eta^{(r)}$.
  For $d \geq 20$, where the 1-RSB bound is (conjectured to be) tight, the optimal choice of parameters indeed uses a trivial $\eta^{(r+1)} = \delta_{\eta^{(r)}}$ with r being 1 in this case.
  For $d \leq 19$, the same choice gives us a 1-RSB upper bound (which is not tight any more). Let us call this the *standard 1-RSB bound*, and, in general, we call an r-RSB bound *standard* if it was obtained by using a deterministic $\eta^{(r)}$ at the start. Then a non-standard bound would use $\eta^{(r+1)}$ (and hence random $\eta_\ell^{(r)}$ variables). Note that a non-standard r-RSB bound is actually a special case of standard $(r + 1)$-RSB bounds in the limit $m_{r+1} \to 0$. So even though it is possible to improve on standard r-step bounds by non-standard r-step bounds, it actually makes more sense to use the extra layer to move to $(r + 1)$-step bounds instead (and use some positive m_{r+1}).

- The full RSB picture is well-understood for the famous Sherrington–Kirkpatrick model [22, 26], where the infimum of the r-RSB bound converges to the free energy as $r \to \infty$. It is reasonable to conjecture that this is the case for the hard-core model as well. There is some progress towards this in [8] where a variational formula is obtained for α_d^*.

2.2 A Specific Choice

The formula in Theorem 2.2 would be hard to work with numerically because it would only give good results for very large λ. So we make a specific choice (similar to the one made in [19, Section 3.2.1] in the case $r = 1$) that may not be optimal but will allow us to use numerical optimization. We consider the limit $\lambda \to \infty$ and $m_1 \to 0$ in a way that $m_1 \log \lambda$ stays constant and x is concentrated on the two-element set $\{0, 1 - 1/\lambda\}$, meaning that $\eta^{(1)}$ is a distribution $q\delta_{1-1/\lambda} + (1 - q)\delta_0$ for some random $q \in [0, 1]$.

For a fixed $\lambda_0 > 1$ let $\log \lambda_0 = m_1 \log \lambda$. First we focus on the expressions $T_1\big(1 + \lambda(1 - x_1) \cdots (1 - x_d)\big)$ and $T_1(1 - x_1 x_2)$. If each $x_\ell \in \{0, 1 - 1/\lambda\}$ was fixed, we would have the following in the limit as $\lambda \to \infty$, $m_1 \to 0$ with $m_1 \log \lambda = \log \lambda_0$:

$$\big(1 + \lambda(1 - x_1) \cdots (1 - x_d)\big)^{m_1} \to \begin{cases} \lambda_0 & \text{if each } x_\ell \text{ is } 0; \\ 1 & \text{otherwise;} \end{cases}$$

$$(1 - x_1 x_2)^{m_1} \to \begin{cases} 1/\lambda_0 & \text{if } x_1 = x_2 = 1 - 1/\lambda; \\ 1 & \text{otherwise.} \end{cases}$$

Therefore, conditioned on

$$\eta_\ell^{(1)} = q_\ell \delta_{1 - 1/\lambda} + (1 - q_\ell)\delta_0$$

for some deterministic $q_1, \ldots, q_d \in [0, 1]$, we get

$$T_1\big(1 + \lambda(1 - x_1) \cdots (1 - x_d)\big) \to 1 + (\lambda_0 - 1)(1 - q_1) \cdots (1 - q_d);$$
$$T_1(1 - x_1 x_2) \to 1 - (1 - 1/\lambda_0)q_1 q_2.$$

In the resulting formula the randomness in layer 1 disappears along with the Parisi parameter m_1. After re-indexing ($k \to k - 1$) we get the following corollary.

Corollary 2.3 *Let $\lambda_0 > 1$ and $0 < m_1, \ldots, m_{r-1} < 1$. Furthermore, fix a deterministic $\pi^{(r-1)} \in \mathcal{P}^{r-1}$ and take d independent copies of recursive sampling:*

$$\pi^{(r-1)} \to \pi_\ell^{(r-2)} \to \cdots \to \pi_\ell^{(1)} \to q_\ell \quad (\ell = 1, \ldots, d).$$

We define the conditional expectations $\mathbb{E}_k$ and the corresponding T_k as before, w.r.t. this new system of random variables. Then

$$\alpha_d^* m_1 \cdots m_{r-1} \log \lambda_0 \leq \log T_{r-1}\big(1 + (\lambda_0 - 1)(1 - q_1) \cdots (1 - q_d)\big)$$
$$- \frac{d}{2} \log T_{r-1}\big(1 - (1 - 1/\lambda_0)q_1 q_2\big).$$

Proof For a formal proof one needs to define an $\eta^{(r)} = \eta_\lambda^{(r)} \in \mathcal{P}^r$ for the fixed $\pi^{(r-1)}$ and any given λ such that the corresponding $\eta_\ell^{(1)}$ is distributed as $q_\ell \delta_{1 - 1/\lambda} + (1 - q_\ell)\delta_0$. Then Theorem 2.2 can be applied and we get the new formula in the limit. $\square$

2.3 Discrete Versions

In our numerical computations we will use the bound of Corollary 2.3 in the special case when each distribution is discrete.

For $r = 1$ we have a deterministic q and we get back (2), while $r = 2$ gives (3).

Let $r = 3$. For any $\lambda_0 > 1$, $0 < m_1, m_2 < 1$, $p_i \geq 0$ with $\sum p_i = 1$, $p_{i,j} \geq 0$ with $\sum_j p_{i,j} = 1$ for every fixed i, and $q_{i,j} \in [0, 1]$ we get that

$$\alpha_d^* m_1 m_2 \log(\lambda_0) \leq \log R^{\text{star}} - \frac{d}{2} \log R^{\text{edge}}, \text{ where}$$

$$R^{\text{star}} = \sum_{i_1} \cdots \sum_{i_d} \left(\prod_{\ell=1}^d p_{i_\ell} \right) \left(\sum_{j_1} \cdots \sum_{j_d} \left(\prod_{\ell=1}^d p_{i_\ell, j_\ell} \right) \left(1 + (\lambda_0 - 1) \prod_{\ell=1}^d (1 - q_{i_\ell, j_\ell}) \right)^{m_1} \right)^{m_2};$$

$$R^{\text{edge}} = \sum_{i_1} \sum_{i_2} p_{i_1} p_{i_2} \left(\sum_{j_1} \sum_{j_2} p_{i_1,j_1} p_{i_2,j_2} \left(1 - (1 - 1/\lambda_0) q_{i_1,j_1} q_{i_2,j_2} \right)^{m_1} \right)^{m_2}.$$

For a general $r \geq 1$, we will index our parameters p_s, q_s with sequences $s = \left(s^{(1)}, \ldots, s^{(k)} \right)$ of length $|s| = k \leq r - 1$. We denote the empty sequence (of length 0) by $\emptyset$. Furthermore, we write $s' \succ s$ if s' is obtained by adding an element to the end of s, that is, $|s'| = |s| + 1$ and the first $|s|$ elements coincide.

Now let S be some set of sequences of length at most $r - 1$ such that $\emptyset \in S$. We partition S into two parts $S_{\leq r-2} \cup S_{r-1}$ based on whether the length of the sequence is at most $r - 2$ or exactly $r - 1$, respectively.

Now the discrete version of the r-RSB bound has the following parameters:

- $\lambda_0 > 1$;
- $0 < m_1, \ldots, m_{r-1} < 1$;
- $p_s \geq 0$, $s \in S$, satisfying

$$\sum_{s' \succ s} p_{s'} = 1 \text{ for each } s \in S_{\leq r-2};$$

- $q_s \in [0, 1]$, $s \in S_{r-1}$.

Now we define the distribution $\pi^{(r-1)} \in \mathcal{P}^{r-1}$ corresponding to the parameters p_s, q_s. Set

$$\pi_s := q_s \in [0, 1] \text{ for any } s \in S_{r-1},$$

and then, recursively for $k = r - 2, r - 3, \ldots, 1, 0$, for a sequence s of length $|s| = k$ let

$$\pi_s := \sum_{s' \succ s} p_{s'} \delta_{\pi_{s'}} \in \mathcal{P}^{r-1-k}.$$

We want to use Corollary 2.3 with $\pi^{(r-1)} := \pi_\emptyset \in \mathcal{P}^{r-1}$. The obtained bound can be expressed as follows.

For any d-tuple $s_1, \ldots, s_d$ of sequences of length $r - 1$, set

$$R^{\text{star}}_{s_1,\ldots,s_d} := 1 + (\lambda_0 - 1)(1 - q_{s_1'}) \cdots (1 - q_{s_d'}), \tag{4}$$

and then, recursively for $k = r - 1, r - 2, \ldots, 1$, for any d-tuple $s_1, \ldots, s_d$ of sequences of length $k - 1$ let

$$R^{\text{star}}_{s_1,\ldots,s_d} := \sum_{s_1' \succ s_1} \cdots \sum_{s_d' \succ s_d} p_{s_1'} \cdots p_{s_d'} \left(R^{\text{star}}_{s_1',\ldots,s_d'} \right)^{m_{r-k}}. \tag{5}$$

Similarly, for any pair s_1, s_2 of sequences of length $r - 1$, set

$$R^{\text{edge}}_{s_1,s_2} := 1 - (1 - 1/\lambda_0) q_{s_1'} q_{s_2'},$$

and then, recursively for $k = r - 1, r - 2, \ldots, 1$. for any pair s_1, s_2 of sequences of length $k - 1$, let

$$R^{\text{edge}}_{s_1,s_2} := \sum_{s_1' \succ s_1} \sum_{s_2' \succ s_2} p_{s_1'} p_{s_2'} \left(R^{\text{edge}}_{s_1',s_2'} \right)^{m_{r-k}}.$$

Then the bound is

$$\alpha_d^* m_1 \ldots m_{r-1} \log(\lambda_0) \leq \log R^{\text{star}}_{\emptyset,\ldots,\emptyset} - \frac{d}{2} \log R^{\text{edge}}_{\emptyset,\emptyset}. \tag{6}$$

Table 4 Our 2-RSB bounds for α_3^* using n atoms	[n]	2-RSB bound
	[2]	0.45080997599102
	[3]	0.45079057802543
	[4]	0.45078995066987
	[5]	0.45078993616987
	[6]	0.45078993583363
	[7]	0.45078993582594
	[11]	0.45078993582525
	[32]	0.45078993582510

Remark 2.4 Normally we fix integers $n_1, \ldots, n_{r-1} \geq 2$ and assume that the k-th elements of our sequences come from the set $\{1, \ldots, n_k\}$. This way the number of free parameters (after taking the sum restrictions on the parameters p_s into account) is

$$(r - 1) + 2n_1 n_2 \cdots n_{r-1}. \tag{7}$$

In the tables of Sect. 3.1 and the Appendix we will refer to such a parameter space as $[n_1, \ldots, n_{r-1}]$.

3 Numerical Optimization

3.1 Numerical Results

Our starting point was the observation in [5] that the 1-RSB formula for α_d^* "is stable towards more steps of replica symmetry breaking" only for $d \geq 20$, so it should not be exact for $d \leq 19$. Therefore the 2-RSB bound in Corollary 2.3 ought to provide an improved upper bound for some choice of $\lambda_0, m_1, \pi^{(1)}$. The optimal $\pi^{(1)}$ may be continuous. Can we achieve significant improvement on the 1-RSB bound even by using some atomic measure $\pi^{(1)} = \sum_{i=1}^n p_i \delta_{q_i}$? In other words, can we find good substituting values for the parameters p_i, q_i of the discrete version (3) using numerical optimization? We were skeptical because we may not be able to use a large enough n to get a good atomic approximation of the optimal $\pi^{(1)}$. Surprisingly, based on our findings it appears that even a small number of atoms may yield close-to-optimal bounds. Table 4 shows our best 2-RSB bounds for $d = 3$ and for different values of n.

Of course, we do not know what the true infimum of the bound in Corollary 2.3 is, but our bounds seem to stabilize very quickly as we increase the number of atoms (n). Also, we experimented with various other approaches that would allow for better approximations of continuous distributions and they all pointed to the direction that the bounds in Table 4 are close to optimal.

This actually gave us the hope that it may not be impossible to get further improvements by considering r-step replica bounds for $r \geq 3$ even though the number of atoms we can use at each layer is indeed very small due to computational capacities. Table 5 shows some bounds we obtained for $d = 3$ and $r \geq 3$ using different parameter spaces (see Remark 2.4).

The dimension of the parameter space (7) depends only on $r, n_1, \ldots, n_{r-1}$ and not on the degree d. However, as we increase d, computing R^{star} (see Sect. 2.3) and its derivative takes longer and we have to settle for using smaller values of r and n_k. At the same time, the 1-RSB formula is presumably getting closer to the truth as we are approaching the phase transition

Table 5 Our 3,4,5-step RSB bounds for α_3^*

r	$[n_1, \ldots, n_{r-1}]$	RSB bound
3	$[5, 4]$	0.45078601768
	$[8, 3]$	0.45078601734
	$[8, 4]$	0.45078601720
4	$[6, 2, 2]$	0.45078537162
	$[5, 3, 2]$	0.45078534630
	$[8, 2, 2]$	0.45078534531
5	$[4, 2, 2, 2]$	0.45078520944

between $d = 19$ and $d = 20$. Nevertheless, we tried to achieve as much improvement as we could for each degree $d = 3, \ldots, 19$. See the Appendix for results for $d \geq 4$.

3.2 Implementation

3.2.1 Efficient Computation

According to (6), our RSB upper bound for α_d^* reads as

$$\frac{\log R_{\emptyset,\ldots,\emptyset}^{\text{star}} - \frac{d}{2} \log R_{\emptyset,\emptyset}^{\text{edge}}}{m_1 \ldots m_{r-1} \log(\lambda_0)}, \tag{8}$$

where $R_{\emptyset,\ldots,\emptyset}^{\text{star}}$ and $R_{\emptyset,\emptyset}^{\text{edge}}$ were defined recursively through $r - 1$ steps, each step involving a multifold summation, see Sect. 2.3 for details. So our task is to minimize (8) as a function of the parameters. During optimization the function and its partial derivatives need to be evaluated at a large number of locations. So it was crucial for us to design program codes that compute them efficiently. Instead of trying to do the summations using `for` loops, the idea is to utilize the powerful array manipulation tools of the Python library NumPy. In particular, one can efficiently perform element-wise calculations or block summations on the multidimensional arrays of NumPy.

First we show how $R_{\emptyset,\ldots,\emptyset}^{\text{star}}$ can be obtained using such tools. Recall that S_k contains sequences of length k and we have a parameter p_s for any $s \in S_1 \cup \cdots \cup S_{r-1}$ and q_s for any $s \in S_{r-1}$. In particular, in the $[n_1, \ldots, n_{r-1}]$ setup we have $|S_k| = n_1 \cdots n_k$. We do the following steps.

- v_k: vector of length $|S_k|$ consisting of p_s, $s \in S_k$ ($1 \leq k \leq r - 1$).
- P_k: d-dimensional array of size $|S_k| \times \cdots \times |S_k|$ obtained by "multiplying" d copies of v_k. (Each element of P_k is a product $p_{s_1} \cdots p_{s_d}$ for some $s_1, \ldots, s_d \in S_k$.)
- M_{r-1}: d-dimensional array of size $|S_{r-1}| \times \cdots \times |S_{r-1}|$ obtained by "multiplying" d copies of the vector consisting of $1 - q_s$, $s \in S_{r-1}$, then multiply each element by $\lambda_0 - 1$ and add 1; cf. (4).
- Then, recursively for $k = r - 1, r - 2, \ldots, 1$, given the $|S_k| \times \cdots \times |S_k|$ array M_k we obtain M_{k-1} as follows: we raise M_k to the power of m_{r-k} and multiply by P_k (both element-wise), and perform a block summation: in the $[n_1, \ldots, n_{r-1}]$ setup we divide the array into $n_k \times \cdots \times n_k$ blocks and replace each with the sum of the elements in the block; cf. (5).
- At the end M_0 will have a single element equal to $R_{\emptyset,\ldots,\emptyset}^{\text{star}}$.

One can compute $R_{\emptyset,\dots,\emptyset}^{\text{edge}}$ similarly, using two-dimensional arrays this time.

Note that during the computation of $R_{\emptyset,\dots,\emptyset}^{\text{star}}$ all the d-dimensional arrays are invariant under any permutation of the d axes. This means that the same products appear in many instances, hence the same calculations are repeated many times in the approach above. However, typically we get plenty of compensation in efficiency due to the fact that all the calculations can be done in one sweep using powerful array tools. Nevertheless, when the degree d gets above 7, we do use another approach in the 2-RSB setting d [n]. In advance, we create a list containing all partitions of d into the sum of n nonnegative integers $d = a_1 + \cdots + a_n$. We also store the corresponding multinomial coefficients $\binom{d}{a_1,\dots,a_n}$ in a vector. Then, at each function call, we go through the list of partitions and compute

$$p_1^{a_1} \cdots p_n^{a_n} \left(1 + (\lambda_0 - 1)(1 - q_1)^{a_1} \cdots (1 - q_n)^{a_n} \right)^{m_1},$$

storing the values in a vector. Then we simply need to take the dot product with the precalculated vector containing the multinomial coefficients.

In both approaches computing the partial derivatives with respect to the parameters (λ, m_k, p_s, q_s) is more involved but can be done using similar techniques (array manipulations and partitioning, respectively). As an example, we show how we can compute $\partial R_{\emptyset,\dots,\emptyset}^{\text{star}}/\partial p_s$ in the first approach. For a given $1 \le \ell \le r-1$ we will do this for all $s \in S_\ell$ at once, resulting in a vector of length $|S_\ell|$ consisting of the partial derivatives w.r.t. each p_s, $s \in S_\ell$. We will use again the vectors v_k and the arrays P_k, M_k obtained during the computation of $R_{\emptyset,\dots,\emptyset}^{\text{star}}$.

- P_k': d-dimensional array of size $|S_k| \times \cdots \times |S_k|$ obtained by "multiplying" the all-ones vector of length $|S_k|$ and $d-1$ copies of v_k.
- D_k: d-dimensional array of size $|S_k| \times \cdots \times |S_k|$ obtained by (element-wise) raising M_k to the power of $m_{r-k} - 1$ and multiplying by m_{r-k} and by P_k.
- For a given $1 \le \ell \le r-1$ we start with M_ℓ, raise it to the power of $m_{r-\ell}$ and multiply it by P_ℓ' (both element-wise) and perform a block summation for blocks of size $1 \times n_\ell \times \cdots \times n_\ell$, resulting in an $|S_\ell| \times |S_{\ell-1}| \times \cdots \times |S_{\ell-1}|$ array that we denote by $M_{\ell-1}'$.
- Then, recursively for $k = \ell-1, \ell-2, \dots, 1$, given the $|S_\ell| \times |S_k| \times \cdots \times |S_k|$ array M_k' we obtain M_{k-1}' as follows: we "stretch" D_k so that its first axis has length $|S_\ell|$ by repeating each element $|S_\ell|/|S_k|$ times (along that first axis) to get an $|S_\ell| \times |S_k| \times \cdots \times |S_k|$ array, which we multiply element-wise by M_k', and perform a block summation for blocks of size $1 \times n_k \times \cdots \times n_k$.
- At the end we get the array M_0' of size $|S_\ell| \times 1 \times \cdots \times 1$. We simply need to multiply its elements by d to get the partial derivatives w.r.t. p_s, $s \in S_\ell$.

3.2.2 Local Optimization

Given a differentiable multivariate function, *gradient descent* means that at each step we move in the opposite direction of the gradient at the current point, and thus (hopefully) converging to a local minimum of the function. This is a very natural strategy because we have the steepest initial descent in that direction. There are other standard iterative algorithms that also use the gradient (i.e., the vector consisting of the partial derivatives). They can make more sophisticated steps because they take previous gradient evaluations into account as well, resulting in a faster convergence to a local minimum. Since we have complicated functions for which gradient evaluations are computationally expensive, it is important for us to reach a local optimum in as few iterations as possible. Specifically, we used the *conjugate gradient* and the *Broyden–Fletcher–Goldfarb–Shanno algorithm*, which are both implemented in the Python library SciPy.
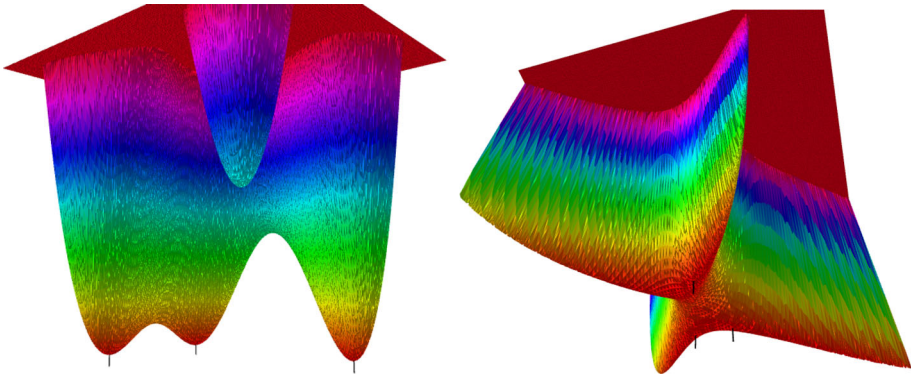
Fig. 1 Plots of our 2-RSB bound over two-dimensional sections. The black ticks mark the local minima. We cut the function at a certain height

With efficient gradient evaluation and fast-converging optimization at our disposal, we were able to find local optima. However, we were surprised to see that, depending on the starting point, these algorithms find a large number of different local minima of the RSB formulas. This is due to our parameterization: we only consider discrete measures with a fixed number of atoms, and the atom locations are included among the parameters.[4] (This is what allowed us to tackle the problem numerically but it also makes the function behave somewhat chaotically.)

It is hard to get a good picture of the behavior of a function of so many variables. To give some idea, in Fig. 1 we plotted the 2-RSB bound for $d = 3$ [$n = 5$] over two-dimensional sections. In both cases we chose three local minima and took the plane H going through them and plotted the function over H. (Note that the left one appears to have a fourth local minimum. However, it is only a local minimum for the two-dimensional restriction of the function and it can actually be locally improved when we are allowed to use all dimensions.)

Many of these local minima have very similar values. It appears that one would basically need to check them all in order to find the one that happens to be the global minimum (for the given number of atoms). So our strategy is to simply start local optimization from various (random) points to eventually get a strong bound. This seems to work well as long as there are not too many local minima.

3.2.3 Basin Hopping

As the dimension of the parameter space grows, we start to see a landscape with a huge number of local minima and our chance for picking a good starting point becomes extremely slim. Instead, when we get to a local minimum (i.e., the bottom of a "basin"), we may try to "hop out" of the basin by applying a small perturbation of the variables. After a new round of local optimization, we end up at the bottom of another basin. If the function value decreases compared to the previous basin, we accept this step. If not, then we make a random decision of acceptance/rejection with a probability based on the difference of the values. Such a basin hopping algorithm randomly travels through local minima, with a slight preference

---

[4] Even in models where the Parisi functional is known to be convex, as in the SK model [1], parameterizing with atom locations (as opposed to with the measure itself) changes the notion of convexity and may lead to functions that are far from convex.

for smaller values. (This preference should not be too strong, though, as we have to allow enough leeway for this random travel.) This approach led to the discovery of our best bounds for $d = 3$. We mention that in the case of our 5-RSB bound the basin hopping algorithm was running for days.

3.2.4 Avoiding Lower-Depth Minima

There is one more subtlety we have to pay attention to, especially when $r \geq 3$. The fact that the r-RSB formula contains the $(r - 1)$-RSB as a special case means that the optimization has the tendency to converge to such "lower-depth" local minima (on the boundary of the parameter space). So it is beneficial to distort the target function in some way in order to force the $r - 1$ Parisi parameters to stay away from the boundary. That is, we need to add a penalty term to our function based on the distance of each m_k from 1. Once the function value is sufficiently small, we can continue the optimization with the original (undistorted) function.

4 One-Step RSB Revisited

It is possible to improve the previous best bounds even within the framework of the $r = 1$ case of the interpolation method. Recall that Theorem 2.2 gives the following bound in this case:

$$\alpha_d^* m \log \lambda \leq \log \mathbb{E}\big(1 + \lambda(1 - x_1) \cdots (1 - x_d)\big)^m - \frac{d}{2} \log \mathbb{E}(1 - x_1 x_2)^m,$$

where $x_1, \ldots, x_d$ are IID from some fixed distribution η on [0, 1]. If we use $\eta = q\delta_{1-1/\lambda} + (1-q)\delta_0$ and take the limit $m \to 0$, $\lambda \to \infty$ with $m \log \lambda = \lambda_0$, then we get (2) as explained in Sect. 2.2 for general r. Optimizing (2) leads to what we refer to as the 1-RSB bound throughout the paper. In this section we show how one can improve on (2) for $d \leq 19$ by considering a more sophisticated η. We will refer to the obtained bounds as 1^+-RSB bounds. Although this approach is generally inferior to 2-RSB bounds, it is computationally less demanding. In fact, for degrees $d = 17, 19$ we could only perform the 2-RSB optimization with $n_1 = 2$ and the obtained bound was actually worse than the 1^+-RSB bound outlined below.

For the sake of simplicity we start with a choice of η only slightly more general than the original one: let η have three atoms at the locations

$$1 - \frac{1}{\lambda^0} = 0; \quad 1 - \frac{1}{\lambda^{1/2}}; \quad 1 - \frac{1}{\lambda^1}.$$

We denote the measures of these atoms by $q_0, q_{1/2}, q_1 \geq 0$, where $q_0 + q_{1/2} + q_1 = 1$; i.e.,

$$\eta = q_0 \delta_0 + q_{1/2} \delta_{1-1/\sqrt{\lambda}} + q_1 \delta_{1-1/\lambda}.$$

Note that the original choice corresponds to the case $q_{1/2} = 0$.

As before, we let $m \to 0$, $\lambda \to \infty$ with $m \log \lambda = \lambda_0$, which leads to the following bound:

$$\alpha_d^* \log(\lambda_0) \leq \log S - \frac{d}{2} \log E, \tag{9}$$

where

$$S = 1 + \left(\sqrt{\lambda_0} - 1\right) d q_0^{d-1} q_{1/2} + \left(\lambda_0 - 1\right) q_0^d;$$

$$E = 1 + (1/\sqrt{\lambda_0} - 1)(1 - q_0)^2 + (1/\lambda_0 - 1/\sqrt{\lambda_0})q_1^2.$$

Substituting $q_1 = 1 - q_0 - q_{1/2}$, we have three remaining parameters: $\lambda_0, q_0, q_{1/2}$. Setting the partial derivatives of the right-hand side w.r.t. q_0 and $q_{1/2}$ to 0, we get that

$$0 = \partial_{q_0}\left(\log S - \frac{d}{2}\log E\right) = \frac{\partial_{q_0}S}{S} - \frac{d}{2}\frac{\partial_{q_0}E}{E},$$

and similarly for $\partial_{q_{1/2}}$. It follows that for the optimal choice of parameters we have

$$\frac{S}{E} = \frac{\frac{1}{d}\partial_{q_0}S}{\frac{1}{2}\partial_{q_0}E} = \frac{\frac{1}{d}\partial_{q_{1/2}}S}{\frac{1}{2}\partial_{q_{1/2}}E}.$$

One can easily compute these partial derivatives to conclude that

$$\frac{S}{E} = \frac{q_0^{d-1}}{\frac{1}{\lambda_0}(1 - q_0 - q_{1/2})} = \frac{(d-1)q_0^{d-2}q_{1/2} + (\sqrt{\lambda_0} + 1)q_0^{d-1}}{\frac{1}{\sqrt{\lambda_0}}(1 - q_0) + \frac{1}{\lambda_0}(1 - q_0 - q_{1/2})}.$$

The second equality gives

$$\sqrt{\lambda_0} = (d-1)\frac{1 - q_0 - q_{1/2}}{q_0},$$

which turns the first equality into

$$q_{1/2} = \frac{d-1}{d-2} - \frac{dq_0}{d-1} - \frac{1}{(d-1)(d-2)q_0^{d-2}}.$$

So our bound has one free parameter left (q_0), in which we can easily optimize numerically. For $d = 3$ one gets 0.450851131. This is the simplest way to improve upon the basic 1-RSB bound.

More generally, one can take any measure τ on $[0, 1]$ and define η as the push-forward of τ w.r.t. the mapping $t \mapsto 1 - 1/\lambda^t$. Once again, letting $m \to 0$, $\lambda \to \infty$ with $m \log \lambda = \lambda_0$, we get the following:

$$\alpha_d^* \log(\lambda_0) \le \log\left(\int \lambda_0^{\max(0, 1 - \sum t_\ell)} d\tau^d(t_1, \ldots, t_d)\right)$$
$$- \frac{d}{2}\log\left(\int \lambda_0^{-\min(t_1, t_2)} d\tau^2(t_1, t_2)\right).$$

For any fixed λ_0, an optimal τ must satisfy a simple fixed point equation involving the convolution power $\tau^{*(d-1)}$. For div $\in \mathbb{N}$ one can divide $[0, 1]$ into div many intervals and search among atomic measures τ with atom locations at i/div, $i = 0, 1, \ldots, \text{div}$. It is possible to numerically solve the fixed point equation by an iterative algorithm. Then it remains to tune the parameter λ_0. We computed these 1^+-RSB bounds for div $= 1, 2, 4, 8, \ldots, 1024$. Note that div $= 1$ corresponds to the original 1-RSB, while div $= 2$ gives (9). Table 6 shows the results for $d = 3$.

Table 6 Our 1^+-RSB bounds for α_3^*

div	1^+-RSB bound
1	0.45085965358
2	0.45085113089
4	0.45084699561
8	0.45084570075
16	0.45084535605
32	0.45084526847
64	0.45084524648
128	0.45084524098
256	0.45084523960
512	0.45084523926
1024	0.45084523917
2-RSB	0.45078993583

Note that div $= 1$ corresponds to 1-RSB

Data Availability The datasets needed to recover the claimed bounds of the current paper are available in a Github repository: https://github.com/harangi/rsb/blob/5f2ea09166a1931c3bae5e05afaee13093015b53/examples.sobj

Declarations

Conflicts of interest The author declares that he has no conflict of interest.

5 Appendix 1: Our Best Bounds

Below we list our best r-RSB bounds of α_d^* for each degree $3 \leq d \leq 19$ in the following format: r $[n_1, \ldots, n_{r-1}]$ bound   (see Remark 2.4 for the definition of n_k).

For comparison, we included $r = 1$, that is, the 1-RSB bound from [19] that we improve on.

degree: 3		
1		0.450859654
2	[32]	0.450789936
3	[8, 4]	0.450786018
4	[8, 2, 2]	0.450785346
5	[4, 2, 2, 2]	0.450785210

degree: 4		
1		0.411194564
2	[18]	0.411100755
3	[6, 4]	0.411095101
4	[4, 3, 2]	0.411094131

degree: 5		
1		0.379268170
2	[8]	0.379176250
3	[3, 3]	0.379170372
4	[2, 2, 3]	0.379170310

degree: 6		
1		0.352984549
2	[7]	0.352905514
3	[4, 2]	0.352900232
4	[3, 2, 2]	0.352899485

degree: 7		
1		0.330884354
2	[5]	0.330821477
3	[5, 2]	0.330817014

degree: 8		
1		0.311972567
2	[6]	0.311925387
3	[3, 2]	0.311922227

degree: 9		
1		0.295553902
2	[5]	0.295520273
3	[2, 2]	0.295519497

degree: 10		
1		0.281128003
2	[5]	0.281105186
3	[2, 2]	0.281104953

degree: 11		
1		0.268324856
2	[7]	0.268310124

degree: 12		
1		0.256864221
2	[5]	0.256855205

degree: 13		
1		0.246529415
2	[6]	0.246524236

degree: 14		
1		0.237149865
2	[4]	0.237147193

degree: 15		
1		0.228589175
2	[4]	0.228587914

degree: 16		
1		0.220736776
2	[4]	0.220736278

degree: 17		
1		0.213501935208
1+		0.213501905193

degree: 18		
1		0.206809394782
1+		0.206809390398
2	[2]	0.206809390050

degree: 19		
1		0.2005961242697
1+		0.2005961242567

6 Appendix 2: An Overview of the Interpolation Method

The interpolation method is a rigorous technique to prove upper bounds for the free energy in various models. It has several variants. Originally it was invented by Guerra [15] in the context of the Sherrington–Kirkpatrick spin glass model. In this section we explain the technique for the hard-core model, omitting the technical details and assuming no statistical physics background. We mainly follow the exposition in [2], where the closely related problem of the chromatic number was considered, and [23].

Given a finite graph $G = (V, E)$, the *partition function* of the *hard-core model* is defined as

$$Z_G = Z_{G,\lambda} := \sum_{\sigma \in \{0,1\}^V} \prod_{v \in V} \lambda^{\sigma_v} \prod_{uv \in E} \left(1 - \mathbb{1}\big(\{\sigma_u = \sigma_v = 1\}\big) \right),$$

where $\lambda > 1$ is a parameter often called fugacity. So Z_G counts 0-1 configurations $\sigma = (\sigma_v)_{v \in V}$ on the vertices with no neighboring 1's, that is, $I := \{v \in V : \sigma_v = 1\}$ is an independent set counted with weight $\lambda^{|I|}$. Thus Z_G is simply the sum of these weights for all independent sets[5]. Let $\alpha(G)$ denote the independence number of G (i.e., the size of the largest independent set). Using the simple inequality $Z_{G,\lambda} \geq \lambda^{\alpha(G)}$, one can bound the independence number as follows:

$$\alpha(G) \leq \frac{\log Z_{G,\lambda}}{\log \lambda},$$

which is clearly asymptotically tight for any fixed G as $\lambda \to \infty$.

We are interested in the asymptotic independence ratio α_d^* of the random d-regular graph $\mathbb{G} = \mathbb{G}(N, d)$ as the number of vertices N goes to infinity. It follows from the above that for any λ the *normalized free energy* $F_N := \mathbb{E} \log Z / N$ upper bounds $\alpha_d^* \log \lambda$. More precisely, we have

$$\alpha_d^* \leq \lim_{N \to \infty} \frac{\mathbb{E}_{\mathbb{G}} \log Z_{\mathbb{G},\lambda}}{N \log \lambda}.$$

The method is based on an "interpolating" family of models $G_t, t \in [0, 1]$, with G_0 being our original model (plus a disjoint part), which is then "continuously transformed" into G_1. The key is to prove that the free energy $\mathbb{E} \log Z_{G_t}$ increases as t goes from 0 to 1 by showing that the derivative is nonnegative along the way:

$$\frac{\partial \mathbb{E} \log Z_{G_t}}{\partial t} \geq 0 \quad \forall t \in [0, 1]. \tag{10}$$

We will elaborate on this key part of the proof later in Sect. 6.5. It implies that $\mathbb{E} \log Z_{G_0} \leq \mathbb{E} \log Z_{G_1}$, which will translate to a bound of the form

$$\mathbb{E} \log Z_{\mathbb{G}} \leq \mathbb{E} \log Y - \mathbb{E} \log Y' + o(N), \tag{11}$$

where Y and Y' are partition functions that are easier to handle. Next we will describe the models in detail.

6.1 Variables and Factors

The models have two types of nodes: *variable nodes* and *fields* (corresponding to *local fields* in physics). We assign a variable σ_v to any variable node v that *ranges over* $\{0, 1\}$. When we compute the partition function, the sum runs through all possible configurations $\sigma = (\sigma_v)$ with the weight of a configuration being the product of various (penalty and reward) factors. For example, each $\sigma_v = 1$ is rewarded with a $\lambda > 1$ factor, while an edge between two variable nodes v, v' forbids that $\sigma_v = \sigma_{v'} = 1$, i.e., the factor is

$$(\sigma_v, \sigma_{v'}) \mapsto 1 - \sigma_v \sigma_{v'} = \begin{cases} 1 & \text{if } \sigma_v = 0 \text{ or } \sigma_{v'} = 0; \\ 0 & \text{if } \sigma_v = \sigma_{v'} = 1. \end{cases}$$

A field u does not have a variable, instead there is a probability distribution μ_u on $\{0, 1\}$ assigned to it. In other words, each field u is labelled with a real number $x_u \in [0, 1]$ denoting the probability of 1: $x_u := \mu_u(\{1\})$.

---

[5] In fact, we should also work with a soft version of Z (at some positive temperature), where neighboring 1's are possible but penalized in the partition function. As the temperature goes to zero (i.e., the penalty increases), we get back the hard-core model in the limit. For the sake of simplicity, we describe the interpolation method using the hard-core model but keep in mind that a rigorous treatment would need positive temperatures.

If there is an edge between a variable node v and a field u, then we use the following factor:

$$\sigma_v \mapsto 1 - x_u \sigma_v = \begin{cases} 1 & \text{if } \sigma_v = 0; \\ \mu_u(\{0\}) = 1 - x_u & \text{if } \sigma_v = 1. \end{cases}$$

Finally, for an edge between two fields u,u' we add the following constant factor (that does not depend on σ):

$$1 - x_u x_{u'}.$$

6.2 The Models

Now we are ready to describe the models G_t and the partition functions Y and Y'.

- Each model G_t has N variable nodes and dN fields. A variable node has d half-edges, each may be connected to another half-edge or to a field.
- In G_t there are $(1-t)dN/2$ edges connecting two fields, the remaining tdN fields are connected to half-edges of variable nodes randomly, and the remaining $(1-t)dN$ half-edges are matched randomly, creating $(1-t)dN/2$ edges between variable nodes.
- In particular, at $t = 0$ we get the disjoint union of a random d-regular graph $\mathbb{G}$ (over the variable nodes) and $dN/2$ pairs of fields, each pair connected by an edge. Therefore

$$\log Z_{G_0} = \log Z_{\mathbb{G}} + \log Y',$$

where Y' is the partition function of the $dN/2$ "field edges".

- At the other endpoint $t = 1$, we have N "stars", each containing one variable node connected to d fields. We denote the corresponding partition function by Y and hence write

$$\log Z_{G_1} = \log Y.$$

These are random models. Note that for $Z_{\mathbb{G}}$ the randomness comes purely from the underlying random graph structure, while for Y and Y' it comes from the random labels x_u of the fields u that we will explain next.

In the simplest scenario one fixes a real number $x \in [0, 1]$ and use $x_u = x$ for each u. In this setup Y and Y' are actually deterministic and can be expressed as products (with the terms corresponding to the N stars and $dN/2$ field edges, respectively):

$$Y = \left(1 + \lambda(1-x)^d\right)^N \text{ and } Y' = \left(1 - x^2\right)^{dN/2}.$$

(Note that the model of Y' does not have any variable nodes and the "sum" is simply the product of constant factors.) Plugging these into (11) we get back the replica symmetric bound (1).

More generally, one may choose each x_u independently from a fixed distribution ν on $[0, 1]$. (It is important to use the same ν for Y and Y'.) The resulting partition functions can be factorized again and we get a more general version of the RS bound:

$$\alpha_d^* \log \lambda \leq \int_{[0,1]^d} \log\left(1 + \lambda(1-x_1)\cdots(1-x_d)\right) d\nu(x_1)\cdots d\nu(x_d)$$

$$-\frac{d}{2} \int_{[0,1]^2} \log\left(1 - x_1 x_2\right) d\nu(x_1) d\nu(x_2).$$

Next we explain how a seemingly insignificant modification of the method turns this approach into a much more powerful tool and resulting in replica symmetry breaking bounds.

6.3 A Weighting Scheme

For a countable index set Γ let us fix weights $w_\gamma \geq 0$, $\gamma \in \Gamma$, with $\sum_{\gamma \in \Gamma} w_\gamma = 1$ (essentially a probability distribution on Γ) and an arbitrary collection of random variables $\left(x^\gamma\right)_{\gamma \in \Gamma}$, each x^γ taking values in $[0, 1]$. For each $\gamma \in \Gamma$ we consider a version Y_γ of Y. To this end we need to take independent copies of the collection $\left(x^\gamma\right)$ for all fields u:

$$\left(x_u^\gamma\right)_{\gamma \in \Gamma} \text{ has the same joint distribution as } \left(x^\gamma\right)_{\gamma \in \Gamma}.$$

We set the label of each field u to be x_u^γ and define Y_γ to be the corresponding partition function. We define Y'_γ similarly. Then the following weighted version of (11) is also true:

$$\mathbb{E} \log Z_{\mathbb{G}} \leq \mathbb{E} \log \sum_{\gamma \in \Gamma} w_\gamma Y_\gamma - \mathbb{E} \log \sum_{\gamma \in \Gamma} w_\gamma Y'_\gamma + o(N). \tag{12}$$

This weighted version is (potentially) more general but it seems that we lose the crucial property of factorization for the formulas inside the log. There is, however, a "magical" (random) choice of the coefficients w_γ (based on the so-called *Derrida–Ruelle cascades*) for which we still have factorization provided that the collection $\left(x^\gamma\right)$ is *hierarchically exchangeable*, which notion was introduced in [3].

For a given $r \geq 1$ we use $\Gamma = \mathbb{N}^r$ as the countable index set. For any fixed parameters $0 < m_1, \ldots, m_r < 1$, there exist random weights w_γ such that for any given $\eta^{(r)}$ of Theorem 2.2 we can define the collection $\left(x^\gamma\right)$ in a way that (12) yields the bound in the theorem.

We define $\left(x^\gamma\right)$ using the notations of Sect. 2.1. For any $1 \leq k \leq r$ and any $\gamma_1, \ldots, \gamma_k \in \mathbb{N}$ we will define a random $\eta^{(r-k+1)}(\gamma_1, \ldots, \gamma_k) \in \mathcal{P}^{r-k+1}$. Since we started with a deterministic $\eta^{(r)}$ in Theorem 2.2 (see the remarks after the theorem), in our case each $\eta^{(r)}(\gamma_1) := \eta^{(r)}$ will be the same for $k = 1$. Given $\eta^{(r-k+1)}(\gamma_1, \ldots, \gamma_k) \in \mathcal{P}^{r-k+1}$, we define

$$\eta^{(r-k)}(\gamma_1, \ldots, \gamma_k, \gamma_{k+1}), \ \gamma_{k+1} \in \mathbb{N},$$

to be conditionally independent and distributed as $\eta^{(r-k+1)}(\gamma_1, \ldots, \gamma_k)$. Finally, for each $\gamma = (\gamma_1, \ldots, \gamma_r) \in \mathbb{N}^r$ we sample x^γ from $\eta^{(1)}(\gamma_1, \ldots, \gamma_r)$. Schematically:

$$\eta^{(r)} = \eta^{(r)}(\gamma_1) \ \rightarrow \ \eta^{(r-1)}(\gamma_1, \gamma_2) \ \rightarrow \ \cdots \ \rightarrow \ \eta^{(1)}(\gamma_1, \ldots, \gamma_r) \ \rightarrow \ x^\gamma.$$

Now suppose that we have a function $f : [0, 1]^M \rightarrow \mathbb{R}$. Let us take M independent copies of the above sampling scheme. For each fixed $\gamma \in \mathbb{N}^r$ we plug the M copies of x^γ into f resulting in a random variable V_γ. Then one can choose the weights w_γ randomly in such a way that

$$\mathbb{E} \log \sum_{\gamma \in \mathbb{N}^r} w_\gamma V_\gamma = \mathbb{E} \log T_r\left(V_{(1,\ldots,1)}\right),$$

where T_r is defined analogously to Definition 2.1 [23, Proposition 2].

We will not elaborate on how the weights w_γ need to be chosen for general r. Instead, we focus on the case $r = 1$ which already captures the essence of the method.

6.4 One-Step RSB

In this case we simply have $\Gamma = \mathbb{N}$ and each $\eta^{(1)}(\gamma)$ is the same deterministic distribution $\eta^{(1)} \in \mathcal{P}^1$. In other words, all field labels x_u^γ are IID across all nodes u in all models Y_γ, Y_γ'. Next we define the random weights w_γ.

Definition 6.1 Given a real number $0 < m < 1$, let $\hat{w}_1 \geq \hat{w}_2 \geq \ldots$ be the nonincreasing enumeration of the points generated by a nonhomogeneous Poisson point process on $[0, \infty)$ with intensity function $t \mapsto t^{-1-m}$. The sum $\hat{W} := \sum \hat{w}_\gamma$ is finite almost surely. For $\gamma \in \mathbb{N}$ let

$$w_\gamma := \hat{w}_\gamma / \hat{W}.$$

The distribution of $(w_1, w_2, \ldots)$ is called the *Poisson–Dirichlet distribution*.

In many statistical physics models the relative cluster sizes are believed to behave as the Poisson–Dirichlet distribution for some m. It has the following magical property.

Lemma 6.2 [23, Proposition 1] *For any fixed $0 < m < 1$ let w_γ, $\gamma \in \mathbb{N}$ be the random weights as above. Then for any IID sequence $X_\gamma > 0$ with $\mathbb{E}X_1^2 < \infty$ we have*

$$\mathbb{E} \log \sum_{\gamma=1}^{\infty} w_\gamma X_\gamma = \frac{1}{m} \log \mathbb{E}X_1^m.$$

Note that on the left we take expectation both in w_γ and in X_γ.

Applying the lemma for $X_\gamma = Y_\gamma$ and also for $X_\gamma = Y_\gamma'$, the bound (12) turns into

$$\mathbb{E} \log Z_{\mathbb{G}} \leq \frac{1}{m} \log \mathbb{E}Y_1^m - \frac{1}{m} \log \mathbb{E}(Y_1')^m + o(N)$$

$$= \frac{1}{m} N \log \mathbb{E}\big(1 + \lambda(1 - x_1) \cdots (1 - x_d)\big)^m - \frac{1}{m} \frac{d}{2} N \log \mathbb{E}(1 - x_1 x_2)^m + o(N),$$

where $x_1, \ldots, x_d$ are IID with distribution $\eta^{(1)}$. Hence we indeed get back Theorem 2.2 for $r = 1$.

6.5 Monotonicity of the Free Energy

Now we turn to the final ingredient (the reason why all this provides an upper bound): the fact that the free energy of the model G_t is monotone increasing as t goes from 0 to 1. In other words, the derivative (10) is nonnegative.

In G_t there are three types of edges (based on whether there are 0, 1, or 2 variable nodes among the endpoints) and we defined G_t by prescribing the number of edges for all three types. In fact, it is better to define G_t in a way that there is a small portion of the variable nodes with degree $d - 1$. Intuitively it is clear that we have to compare the effect (on the free energy) of the addition of an edge of each of the three types. (See [2, Section 4.2] for an elegant argument justifying this intuition.)

Suppose that we have any fixed model on N variable nodes with partition function Z, where we distinguish some of the nodes as *cavity nodes* (in our setting they belong to the variable nodes that do not have full degree d but only degree $d - 1$). The number of cavity nodes should be small compared to N but should converge to ∞ as $N \to \infty$. We want to understand the effect (on $\log Z$) of the addition of a new factor to the model. In our case this will be the addition of either one of the three types of edges:

- We choose two cavity nodes uniformly and independently and add an edge between them: resulting in a random partition function Z_{cc}.
- We add two new fields and add an edge between them: resulting in a random partition function Z_{ff}.
- We choose a cavity node uniformly and connect it to a new random field: resulting in a random partition function Z_{cf}.

What we need to prove is that

$$\left(\mathbb{E}\log Z_{cc} - \log Z\right) + \left(\mathbb{E}\log Z_{ff} - \log Z\right) - 2\left(\mathbb{E}\log Z_{cf} - \log Z\right) \le 0. \qquad (13)$$

To incorporate the Replica Symmetry Breaking scenario we will have an additional variable γ: let $\Omega = \{0, 1\}^N \times \Gamma$ where each $\omega = (\sigma_1, \ldots, \sigma_N, \gamma) \in \Omega$ encodes a configuration of N variables σ_i and a *state* γ ranging over a countable set Γ.

Imagine that at a particular stage of the interpolation we see a certain deterministic model. It is actually not important what the model is; the point is that it assigns a weight $\Psi(\omega)$ to each configuration $\omega \in \Omega$. If we normalize these weights with the corresponding partition function $Z = \sum_{\omega \in \Omega} \Psi(\omega)$, then we get a probability distribution on Ω, called the *Boltzmann distribution*. It is a simple fact that adding a new weight factor $\Psi'(\omega)$ to the model changes the free energy $\log Z$ by $\log \mathbb{E}_\omega \Psi'(\omega)$, where $\mathbb{E}_\omega$ means taking expectation w.r.t. the Boltzmann distribution. It follows that

$$\mathbb{E}\log Z_{cc} - \log Z = \mathbb{E}_{c_1,c_2} \log \mathbb{E}_\omega \left(1 - \sigma_{c_1}\sigma_{c_2}\right);$$
$$\mathbb{E}\log Z_{ff} - \log Z = \mathbb{E}_{x_1,x_2} \log \mathbb{E}_\omega \left(1 - x_1^\gamma x_2^\gamma\right);$$
$$\mathbb{E}\log Z_{cf} - \log Z = \mathbb{E}_{c_1,x_1} \log \mathbb{E}_\omega \left(1 - \sigma_{c_1} x_1^\gamma\right),$$

where c_1, c_2 are chosen uniformly and independently from the set $C \subseteq \{1, \ldots, N\}$ of cavities, and $x_1 = (x_1^\gamma)_{\gamma \in \Gamma}$ and $x_2 = (x_2^\gamma)_{\gamma \in \Gamma}$ are two independent collections of random variables with the same joint distribution. Then (13) follows from the following lemma.

Lemma 6.3 *Let X and Y be random $\Omega \to [0, 1]$ functions with independent copies X_1, X_2 and Y_1, Y_2, respectively. Then for any random $\omega \in \Omega$ we have*

$$\mathbb{E}_{X_1,X_2} \log\left(1 - \mathbb{E}_\omega X_1(\omega)X_2(\omega)\right) + \mathbb{E}_{Y_1,Y_2} \log\left(1 - \mathbb{E}_\omega Y_1(\omega)Y_2(\omega)\right)$$
$$\le 2\mathbb{E}_{X,Y} \log\left(1 - \mathbb{E}_\omega X(\omega)Y(\omega)\right).$$

Proof Due to the identity

$$\log(1-x) = -\sum_{\ell=1}^{\infty} \frac{x^\ell}{\ell},$$

it suffices to show for each $\ell \ge 1$ that

$$\mathbb{E}_{X_1,X_2}\left(\mathbb{E}_\omega X_1(\omega)X_2(\omega)\right)^\ell + \mathbb{E}_{Y_1,Y_2}\left(\mathbb{E}_\omega Y_1(\omega)Y_2(\omega)\right)^\ell - 2\mathbb{E}_{X,Y}\left(\mathbb{E}_\omega X(\omega)Y(\omega)\right)^\ell \ge 0, \qquad (14)$$

which can be easily seen to be equivalent to

$$\mathbb{E}_{\omega_1,\ldots,\omega_\ell}\left(\mathbb{E}_X \prod_{i=1}^{\ell} X(\omega_i) - \mathbb{E}_Y \prod_{i=1}^{\ell} Y(\omega_i)\right)^2 \ge 0,$$

where $\omega_1, \ldots, \omega_\ell$ are independent copies of ω.

Indeed, we may rewrite the first term of (14) as

$$
\mathbb{E}_{X_1, X_2} \mathbb{E}_{\omega_1, \ldots, \omega_\ell} \prod_{i=1}^{\ell} X_1(\omega_i) X_2(\omega_i) = \mathbb{E}_{\omega_1, \ldots, \omega_\ell} \left(\mathbb{E}_{X_1} \prod_{i=1}^{\ell} X_1(\omega_i) \right) \left(\mathbb{E}_{X_2} \prod_{i=1}^{\ell} X_2(\omega_i) \right)
$$

$$
= \mathbb{E}_{\omega_1, \ldots, \omega_\ell} \left(\mathbb{E}_X \prod_{i=1}^{\ell} X(\omega_i) \right)^2.
$$

Similar manipulations can be carried out for the two other terms. □

References

1. Auffinger, A., Chen, W.-K.: The parisi formula has a unique minimizer. Commun. Math. Phys. **335**(3), 1429–1444 (2015)
2. Ayre, P., Coja-Oghlan, A., Greenhill, C.: Lower bounds on the chromatic number of random graphs. Combinatorica **42**(5), 617–658 (2022)
3. Austin, T., Panchenko, D.: A hierarchical version of the de Finetti and Aldous-Hoover representations. Probab. Theory Relat. Fields **159**(3), 809–823 (2014)
4. Bayati, M., Gamarnik, D., Tetali, P.: Combinatorial approach to the interpolation method and scaling limits in sparse random graphs. The Annals of Probability **41**(6), 4080–4115 (2013)
5. Barbier, J., Krzakala, F., Zdeborová, Lenka, Zhang, P.: The hard-core model on random graphs revisited. J. Phys. **473**, 012021 (2013)
6. Bollobás, B.: The independence ratio of regular graphs. Proc. Am. Math. Soc. **83**(2), 433–436 (1981)
7. Csóka, E., Gerencsér, B., Harangi, V., Virág, B.: Invariant Gaussian processes and independent sets on regular graphs of large girth. Random Struct. Algorithms **47**(2), 284–303 (2015)
8. Coja-Oghlan, A., Perkins, W.: Spin systems on Bethe lattices. Commun. Math. Phys. **372**(2), 441–523 (2019)
9. Csóka, E.: Independent sets and cuts in large-girth regular graphs (2016)
10. Ding, J., Sly, A., Sun, N.: Maximum independent sets on random regular graphs. Acta Math. **217**(2), 263–340 (2016)
11. Frieze, A.M., Łuczak, T.: On the independence and chromatic numbers of random regular graphs. J. Comb. Theory Ser. B **54**(1), 123–132 (1992)
12. Franz, S., Leone, M.: Replica bounds for optimization problems and diluted spin systems. J. Stat. Phys. **111**(3), 535–564 (2003)
13. Franz, S., Leone, M., Toninelli, F.L.: Replica bounds for diluted non-Poissonian spin systems. J. Phys. A **36**, 10967–10985 (2003)
14. Gamarnik, D., Sudan, M.: Limits of local algorithms over sparse random graphs. Proceedings of the 5-th Innovations in Theoretical Computer Science conference. ACM Special Interest Group on Algorithms and Computation Theory (2014)
15. Guerra, F.: Broken replica symmetry bounds in the mean field spin glass model. Commun. Math. Phys. **233**(1), 1–12 (2003)
16. Hoppen, C.: Properties with graphs of large girth. PhD Thesis, University of Waterloo (2008)
17. Hoppen, C., Wormald, N.: Local algorithms, regular graphs of large girth, and random regular graphs. Combinatorica **38**(3), 619–664 (2018)
18. Kardoš, F., Král, D., Volec, J.: Fractional colorings of cubic graphs with large girth. SIAM J. Discret. Math. **25**(3), 1454–1476 (2011)
19. Lelarge, M., Oulamara, M.: Replica bounds by combinatorial interpolation for diluted spin systems. J. Stat. Phys. **173**(3), 917–940 (2018)
20. McKay, B.D.: Independent sets in regular graphs of high girth. Ars Comb. **23A**, 179–185 (1987)
21. Mézard, M., Parisi, G.: The Bethe lattice spin glass revisited. Eur. Phys. J. B **20**(2), 217–233 (2001)
22. Panchenko, D.: The Sherrington–Kirkpatrick Model. Springer Monographs in Mathematics, Springer, New York (2013)
23. Panchenko, D., Talagrand, M.: Bounds for diluted mean-fields spin glass models. Probab. Theory Relat. Fields **130**(3), 319–336 (2004)

24. Rivoire, O.: Phases vitreuses, optimisation et grandes déviations. Theses, Université Paris Sud—Paris XI, July 2005. les articles devant figurer en appendices ne sont pas inclus (2005)
25. Rahman, M., Virág, B.: Local algorithms for independent sets are half-optimal. Ann. Probab. **45**(3), 1543–1577 (2017)
26. Talagrand, M.: The Parisi formula. Ann. Math. **163**(1), 221–263 (2006)