From Statistical Mechanical Models to Tensor Network Decoding

Christopher Chubb
Université de Sherbrooke

Stat Mech Mapping: arXiv:1809.10704, to appear in AIHPD
with Steve Flammia

Tensor Network decoding: To appear arXiv:2009:?????

Decoders

Passive error correction: physics alone suppress errors

Active error correction: decoder needed to remove error

Two classes of decoder:

- Practical decoders: Speed over accuracy
- Analytic decoders: Accuracy over speed

Decoders

Passive error correction: physics alone suppress errors

Active error correction: decoder needed to remove error

Two classes of decoder:

- Practical decoders: Speed over accuracy
- Analytic decoders: Accuracy over speed

Decoders

Passive error correction: physics alone suppress errors

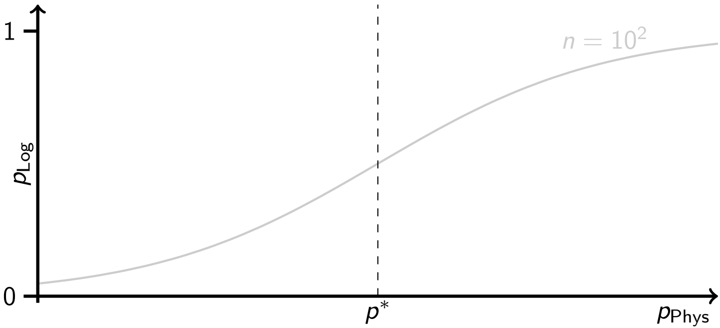
Active error correction: decoder needed to remove error

Two classes of decoder:

- Practical decoders: Speed over accuracy
- **Analytic decoders: Accuracy over speed**

Threshold

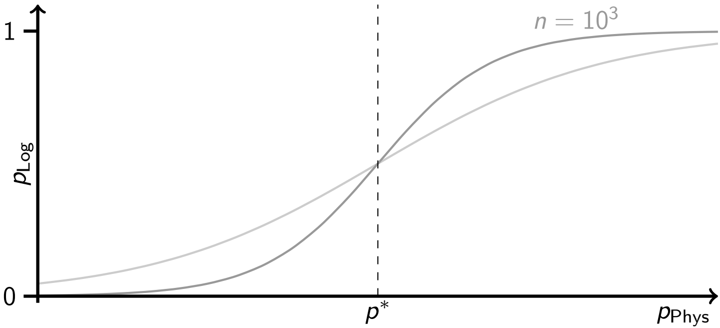
For large system sizes, performance is largely described by the threshold.



The inherent performance of the code can be studied by considering the optimal decoder.

Threshold

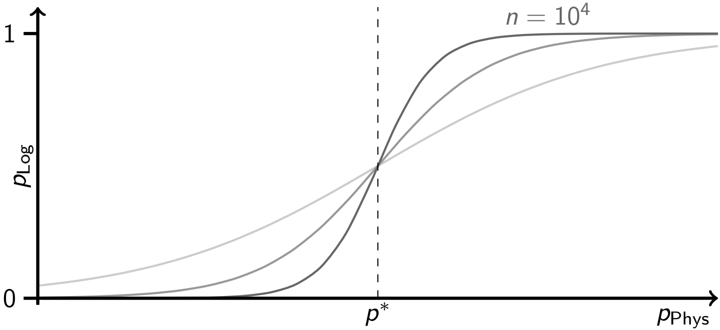
For large system sizes, performance is largely described by the threshold.



The inherent performance of the code can be studied by considering the optimal decoder.

Threshold

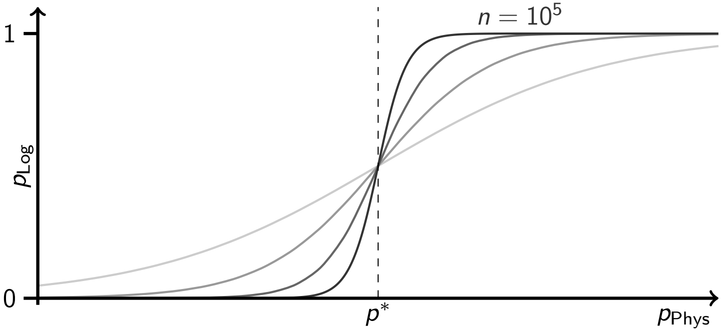
For large system sizes, performance is largely described by the threshold.



The inherent performance of the code can be studied by considering the optimal decoder.

Threshold

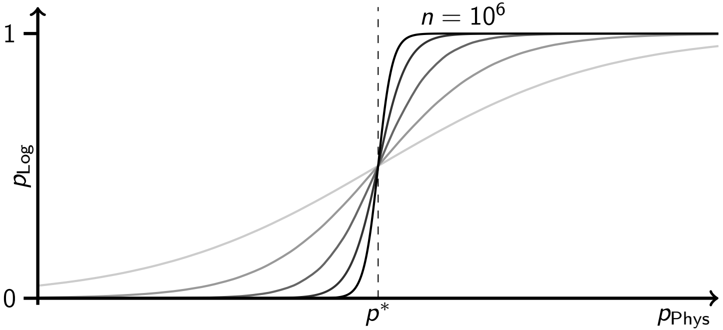
For large system sizes, performance is largely described by the threshold.



The inherent performance of the code can be studied by considering the optimal decoder.

Threshold

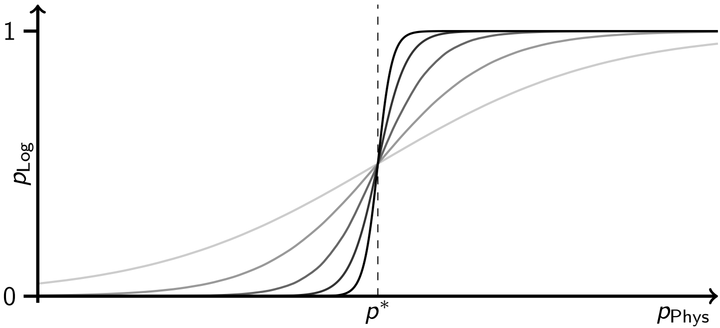
For large system sizes, performance is largely described by the threshold.



The inherent performance of the code can be studied by considering the optimal decoder.

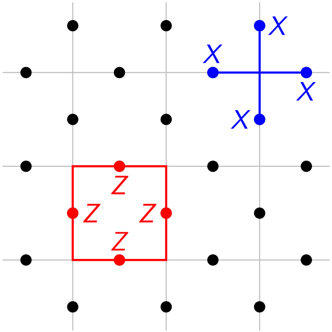
Threshold

For large system sizes, performance is largely described by the threshold.



The inherent performance of the code can be studied by considering the optimal decoder.

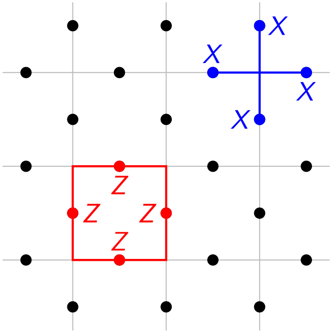
Toric/surface code



The toric code admits a homological interpretation:

- Stabilisers form closed loops
- Logical operators form non-contractible loops
- Errors correspond to open paths
- Syndrome bits corresponds to the ends of paths

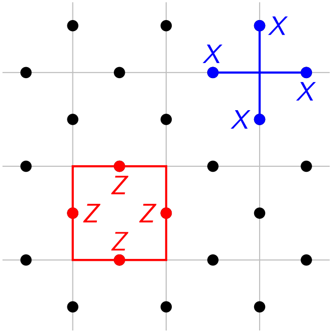
Toric/surface code



The toric code admits a homological interpretation:

- Stabilisers form closed loops
- Logical operators form non-contractible loops
- Errors correspond to open paths
- Syndrome bits corresponds to the ends of paths

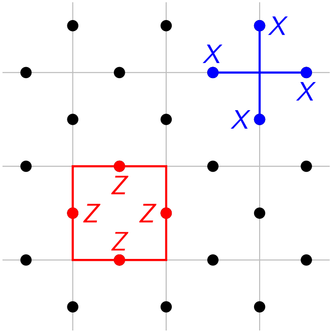
Toric/surface code



The toric code admits a homological interpretation:

- Stabilisers form closed loops
- Logical operators form non-contractible loops
- Errors correspond to open paths
- Syndrome bits corresponds to the ends of paths

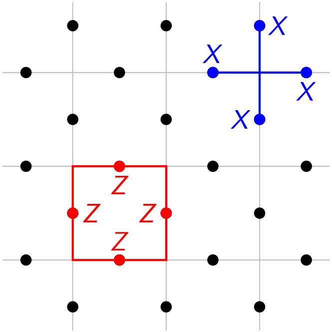
Toric/surface code



The toric code admits a homological interpretation:

- Stabilisers form closed loops
- Logical operators form non-contractible loops
- Errors correspond to open paths
- Syndrome bits corresponds to the ends of paths

Toric/surface code



The toric code admits a homological interpretation:

- Stabilisers form closed loops
- Logical operators form non-contractible loops
- Errors correspond to open paths
- Syndrome bits corresponds to the ends of paths

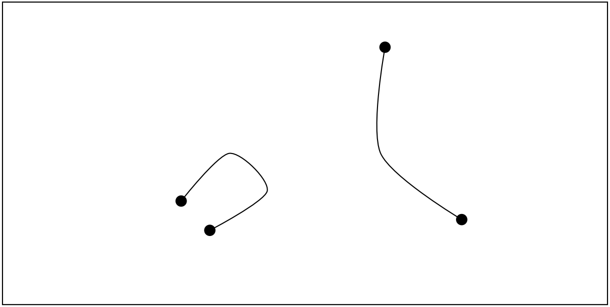
Pair matching decoder

If errors are open paths, why not just close them? This leads to the pair matching decoder.

This decoder is efficient, but its slightly suboptimal for bit-flip and phase-flip errors, and performs badly for other errors.

Pair matching decoder

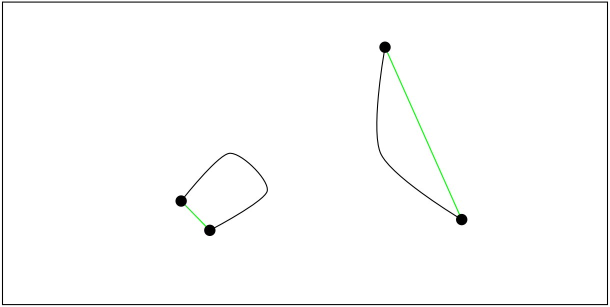
If errors are open paths, why not just close them? This leads to the pair matching decoder.



This decoder is efficient, but its slightly suboptimal for bit-flip and phase-flip errors, and performs badly for other errors.

Pair matching decoder

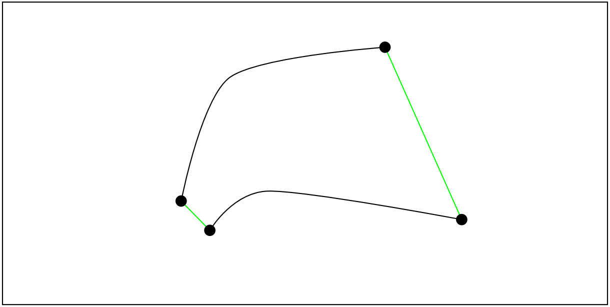
If errors are open paths, why not just close them? This leads to the pair matching decoder.



This decoder is efficient, but its slightly suboptimal for bit-flip and phase-flip errors, and performs badly for other errors.

Pair matching decoder

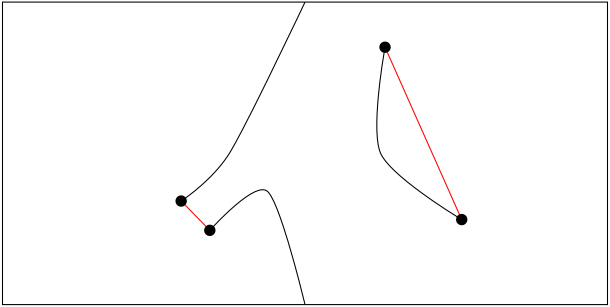
If errors are open paths, why not just close them? This leads to the pair matching decoder.



This decoder is efficient, but its slightly suboptimal for bit-flip and phase-flip errors, and performs badly for other errors.

Pair matching decoder

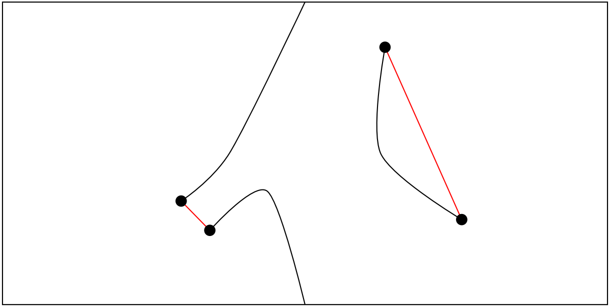
If errors are open paths, why not just close them? This leads to the pair matching decoder.



This decoder is efficient, but its slightly suboptimal for bit-flip and phase-flip errors, and performs badly for other errors.

Pair matching decoder

If errors are open paths, why not just close them? This leads to the pair matching decoder.



This decoder is efficient, but its slightly suboptimal for bit-flip and phase-flip errors, and performs badly for other errors.

Most likely error versus maximum likelihood

Why is pair matching suboptimal? In short, degeneracy.

For quantum codes multiple errors can have the same syndrome. A decoder needs to identify the most likely error <u>class</u>, not the single most likely error.

$E_1, E_2, \ldots$ $\qquad$ $E_3, E_4, \ldots$

Most likely error versus maximum likelihood

Why is pair matching suboptimal? In short, degeneracy.

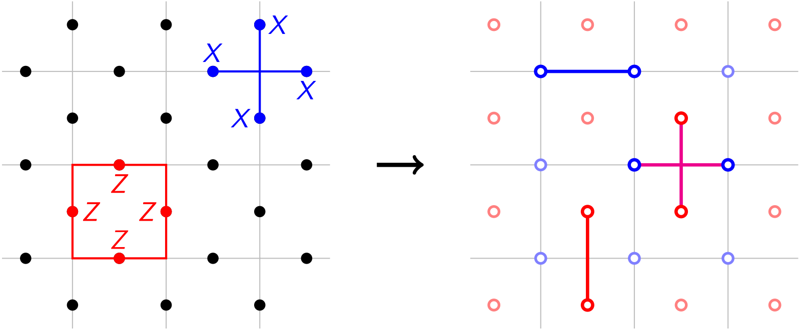
For quantum codes multiple errors can have the same syndrome. A decoder needs to identify the most likely error class, not the single most likely error.

$E_1, E_2, \ldots$

$E_3, E_4, \ldots$

Statistical mechanical mapping

The idea here it to construct a family of statistical mechanical models, whose thermodynamic properties reflect the error correction properties of the code.



Toric code → Eight-vertex model

This will allow us to use the analytic and numerical tools developed to study stat mech systems to study quantum codes.

Statistical mechanical mapping

Stabiliser code & Pauli noise	$\longrightarrow$	Disordered statistical mechanical model
Threshold	$\longleftrightarrow$	Phase transition
Decoding	$\longleftrightarrow$	Calculating partition functions

Allows us to reappropriate techniques for studying stat. mech. systems to study quantum codes, e.g.

Threshold approximation	$\longleftarrow$	Monte Carlo simulation
Optimal decoding	$\longleftarrow$	Partition function calculation

Statistical mechanical mapping

Stabiliser code & Pauli noise	$\longrightarrow$	Disordered statistical mechanical model
Threshold	$\longleftrightarrow$	Phase transition
Decoding	$\longleftrightarrow$	Calculating partition functions

Allows us to reappropriate techniques for studying stat. mech. systems to study quantum codes, e.g.

Threshold approximation	$\longleftarrow$	Monte Carlo simulation
Optimal decoding	$\longleftarrow$	Partition function calculation

Statistical mechanical mapping

Stabiliser code & Pauli noise	$\longrightarrow$	Disordered statistical mechanical model
Threshold	$\longleftrightarrow$	Phase transition
Decoding	$\longleftrightarrow$	Calculating partition functions

Allows us to reappropriate techniques for studying stat. mech. systems to study quantum codes, e.g.

Threshold approximation	$\longleftarrow$	Monte Carlo simulation
Optimal decoding	$\longleftarrow$	Partition function calculation

Stabiliser codes and Pauli noise

For qubits, the Paulis $\mathcal{P} := \{I, X, Y, Z\}$ are defined

$$I := \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}, \quad X := \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix}, \quad Y := \begin{pmatrix} 0 & -i \\ i & 0 \end{pmatrix}, \quad Z := \begin{pmatrix} 1 & 0 \\ 0 & -1 \end{pmatrix}.$$

We will be considering stabiliser codes, which are specified by an Abelian subgroup of the Paulis $\mathcal{S}$, and whose code space $\mathcal{C}$ is the joint $+1$ eigenspace,

$$\mathcal{C} = \left\{ |\psi\rangle \,\middle|\, S\,|\psi\rangle = |\psi\rangle\,, \forall S \in \mathcal{S} \right\}.$$

Any two errors which differ by a stabiliser are logically equivalent, so the logical classes of errors are

$$\overline{E} := \{ES | S \in \mathcal{S}\}$$

Independent case: Hamiltonian

Let $[\![A, B]\!]$ be the scalar commutator of two Paulis, such that $AB =: [\![A, B]\!]\, BA$.

For a stabiliser code generated by $\{S_k\}_k$, and an error Pauli E, the (disordered) Hamiltonian H_E is defined

$$H_E(\vec{s}) := -\sum_i \sum_{\sigma \in \mathcal{P}_i} \overbrace{J_i(\sigma)}^{\text{Coupling}} \overbrace{[\![\sigma, E]\!]}^{\text{Disorder}} \overbrace{\prod_{k : [\![\sigma, S_k]\!] = -1} s_k}^{\text{DoF}}$$

for $s_k = \pm 1$, and coupling strengths $J_i(\sigma) \in \mathbb{R}$.

Take-aways:

- Ising-type, with interactions corresponding to single-site Paulis σ
- Disorder E flips some interactions (Ferro $\leftrightarrow$ Anti-ferro)
- Local code $\implies$ local stat. mech. model

Independent case: Hamiltonian

Let $[\![A, B]\!]$ be the scalar commutator of two Paulis, such that $AB =: [\![A, B]\!]\, BA$.

For a stabiliser code generated by $\{S_k\}_k$, and an error Pauli E, the (disordered) Hamiltonian H_E is defined

$$H_E(\vec{s}) := -\sum_i \sum_{\sigma \in \mathcal{P}_i} \overbrace{J_i(\sigma)}^{\text{Coupling}} \overbrace{[\![\sigma, E]\!]}^{\text{Disorder}} \overbrace{\prod_{k:[\![\sigma, S_k]\!]=-1} s_k}^{\text{DoF}}$$

for $s_k = \pm 1$, and coupling strengths $J_i(\sigma) \in \mathbb{R}$.

Take-aways:

- Ising-type, with interactions corresponding to single-site Paulis σ
- Disorder E flips some interactions (Ferro $\leftrightarrow$ Anti-ferro)
- Local code $\implies$ local stat. mech. model

Independent case: Hamiltonian

Let $[\![A, B]\!]$ be the scalar commutator of two Paulis, such that $AB =: [\![A, B]\!] BA$.

For a stabiliser code generated by $\{S_k\}_k$, and an error Pauli E, the (disordered) Hamiltonian H_E is defined

$$H_E(\vec{s}) := -\sum_i \sum_{\sigma \in \mathcal{P}_i} \overbrace{J_i(\sigma)}^{\text{Coupling}} \overbrace{[\![\sigma, E]\!]}^{\text{Disorder}} \overbrace{\prod_{k : [\![\sigma, S_k]\!] = -1} s_k}^{\text{DoF}}$$

for $s_k = \pm 1$, and coupling strengths $J_i(\sigma) \in \mathbb{R}$.

Take-aways:

- Ising-type, with interactions corresponding to single-site Paulis σ
- Disorder E flips some interactions (Ferro $\leftrightarrow$ Anti-ferro)
- Local code $\implies$ local stat. mech. model

Independent case: Gauge symmetry

$$H_E(\vec{s}) = -\sum_i \sum_{\sigma \in \mathcal{P}_i} J_i(\sigma) [\![\sigma, E]\!] \prod_{k : [\![\sigma, S_k]\!] = -1} s_k$$

Using $[\![A, B]\!] \, [\![A, C]\!] = [\![A, BC]\!]$, we see this system has a gauge symmetry

$$s_k \to -s_k \qquad \text{and} \qquad E \to ES_k.$$

This gauge symmetry will capture the logical equivalence of errors, $Z_E = Z_{ES_k}$.

Independent case: Gauge symmetry

$$H_E(\vec{s}) = -\sum_i \sum_{\sigma \in \mathcal{P}_i} J_i(\sigma) [\![\sigma, E]\!] \prod_{k : [\![\sigma, S_k]\!] = -1} s_k$$

Using $[\![A, B]\!] \, [\![A, C]\!] = [\![A, BC]\!]$, we see this system has a gauge symmetry

$$s_k \to -s_k \qquad \text{and} \qquad E \to ES_k.$$

This gauge symmetry will capture the logical equivalence of errors, $Z_E = Z_{ES_k}$.

Independent case: Nishimori conditon

Suppose we have an independent error model

$$\Pr(E) = \prod_i p_i(E_i),$$

we now want $Z_E = \Pr(\overline{E})$.

Using the gauge symmetry we have that the partition function can be written as a sum stabiliser-equivalent errors

$$Z_E = \sum_{\vec{s}} e^{-\beta H_E(\vec{s})} = \sum_S e^{-\beta H_{ES}(\vec{1})} = \sum_{F \in \overline{E}} e^{-\beta H_F(\vec{1})}.$$

If we select the coupling strength such that $e^{-\beta H_E(\vec{1})} = \Pr(E)$, then $Z_E = \Pr(\overline{E})$ will follow.

Independent case: Nishimori conditon

Suppose we have an independent error model

$$\Pr(E) = \prod_i p_i(E_i),$$

we now want $Z_E = \Pr(\overline{E})$.

Using the gauge symmetry we have that the partition function can be written as a sum stabiliser-equivalent errors

$$Z_E = \sum_{\vec{s}} e^{-\beta H_E(\vec{s})} = \sum_S e^{-\beta H_{ES}(\vec{1})} = \sum_{F \in \overline{E}} e^{-\beta H_F(\vec{1})}.$$

If we select the coupling strength such that $e^{-\beta H_E(\vec{1})} = \Pr(E)$, then $Z_E = \Pr(\overline{E})$ will follow.

Independent case: Nishimori conditon

Suppose we have an independent error model

$$\Pr(E) = \prod_i p_i(E_i),$$

we now want $Z_E = \Pr(\overline{E})$.

Using the gauge symmetry we have that the partition function can be written as a sum stabiliser-equivalent errors

$$Z_E = \sum_{\vec{s}} e^{-\beta H_E(\vec{s})} = \sum_{S} e^{-\beta H_{ES}(\vec{1})} = \sum_{F \in \overline{E}} e^{-\beta H_F(\vec{1})}.$$

If we select the coupling strength such that $e^{-\beta H_E(\vec{1})} = \Pr(E)$, then $Z_E = \Pr(\overline{E})$ will follow.

Independent case: Nishimori condition

We now want to pick our couplings such that $e^{-\beta H_E(\vec{1})} = \Pr(E)$. Expanding this out, we get

$$\sum_i \log p_i(E) = -\sum_i \sum_\sigma \beta J_i(\sigma) \llbracket \sigma, E \rrbracket.$$

Using the Fourier-like orthogonality relation $\frac{1}{4}\sum_\sigma \llbracket \sigma, \tau \rrbracket = \delta_{\tau, I}$, this becomes

Nishimori condition: $\beta J_i(\sigma) = \frac{1}{4} \sum_{\tau \in \mathcal{P}} \log p_i(\tau) \llbracket \sigma, \tau \rrbracket ,$

which implies $e^{-\beta H_E(\vec{1})} = \Pr(E)$, and therefore $Z_E = \Pr(\overline{E})$.

This intrinsically links the error correcting behaviour of the code to the thermodynamic behaviour of the model (along the Nishimori line).

Independent case: Nishimori condition

We now want to pick our couplings such that $e^{-\beta H_E(\vec{1})} = \Pr(E)$. Expanding this out, we get

$$\sum_i \log p_i(E) = -\sum_i \sum_\sigma \beta J_i(\sigma) \llbracket \sigma, E \rrbracket.$$

Using the Fourier-like orthogonality relation $\frac{1}{4}\sum_\sigma \llbracket \sigma, \tau \rrbracket = \delta_{\tau,I}$, this becomes

Nishimori condition: $$\beta J_i(\sigma) = \frac{1}{4} \sum_{\tau \in \mathcal{P}} \log p_i(\tau) \llbracket \sigma, \tau \rrbracket,$$

which implies $e^{-\beta H_E(\vec{1})} = \Pr(E)$, and therefore $Z_E = \Pr(\overline{E})$.

This intrinsically links the error correcting behaviour of the code to the thermodynamic behaviour of the model (along the Nishimori line).

Independent case: Nishimori condition

We now want to pick our couplings such that $e^{-\beta H_E(\vec{1})} = \Pr(E)$. Expanding this out, we get

$$\sum_i \log p_i(E) = -\sum_i \sum_\sigma \beta J_i(\sigma) \llbracket \sigma, E \rrbracket.$$

Using the Fourier-like orthogonality relation $\frac{1}{4}\sum_\sigma \llbracket \sigma, \tau \rrbracket = \delta_{\tau,I}$, this becomes

Nishimori condition: $$\beta J_i(\sigma) = \frac{1}{4} \sum_{\tau \in \mathcal{P}} \log p_i(\tau) \llbracket \sigma, \tau \rrbracket,$$

which implies $e^{-\beta H_E(\vec{1})} = \Pr(E)$, and therefore $Z_E = \Pr(\overline{E})$.

This intrinsically links the error correcting behaviour of the code to the thermodynamic behaviour of the model (along the Nishimori line).

Toric code and the random-bond Ising model

Step 0: Code and noise model

Toric code with iid bit-flips

Step 1: Degrees of freedom

$s_v = \pm 1$ on each vertex v

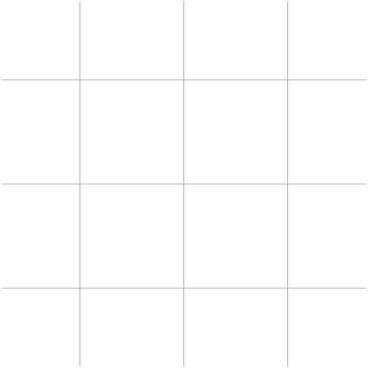
Step 2: Interactions

$$H_I = -\sum_{v \sim v'} J\, s_v s_{v'}$$

Step 3: Disorder

$$H_E = -\sum_{v \sim v'} J e_{vv'}\, s_v s_{v'}$$

where $e_{vv'} = \begin{cases} +1 & E_{vv'} = I, \\ -1 & E_{vv'} = X. \end{cases}$



Toric code and the random-bond Ising model

Step 0: Code and noise model

Toric code with iid bit-flips

Step 1: Degrees of freedom

$s_v = \pm 1$ on each vertex v

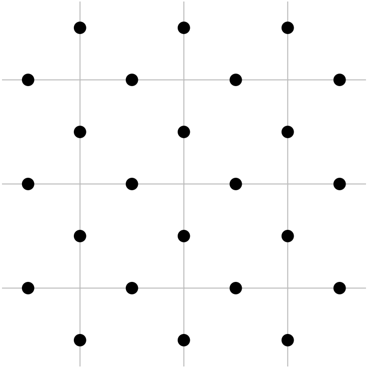
Step 2: Interactions

$$H_I = -\sum_{v \sim v'} J\, s_v s_{v'}$$

Step 3: Disorder

$$H_E = -\sum_{v \sim v'} J e_{vv'}\, s_v s_{v'}$$

$$\text{where } e_{vv'} = \begin{cases} +1 & E_{vv'} = I, \\ -1 & E_{vv'} = X. \end{cases}$$



Toric code and the random-bond Ising model

Step 0: Code and noise model

Toric code with iid bit-flips

Step 1: Degrees of freedom

$s_v = \pm 1$ on each vertex v

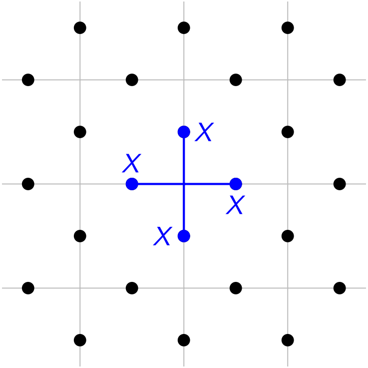
Step 2: Interactions

$$H_I = -\sum_{v\sim v'} J\, s_v s_{v'}$$

Step 3: Disorder

$$H_E = -\sum_{v\sim v'} J e_{vv'}\, s_v s_{v'}$$

$$\text{where } e_{vv'} = \begin{cases} +1 & E_{vv'} = I, \\ -1 & E_{vv'} = X. \end{cases}$$

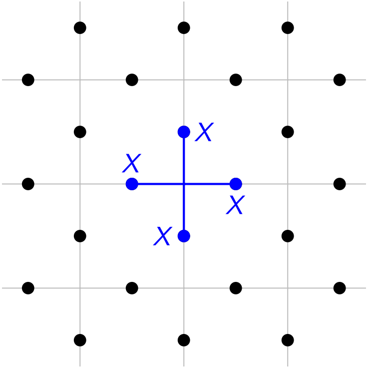


Toric code and the random-bond Ising model

Step 0: Code and noise model

Toric code with iid bit-flips

$\Pr(X_e) = p, \qquad \Pr(I_e) = 1 - p.$



Step 1: Degrees of freedom

$s_v = \pm 1$ on each vertex v

Step 2: Interactions

$$H_I = -\sum_{v \sim v'} J\, s_v s_{v'}$$

Step 3: Disorder

$$H_E = -\sum_{v \sim v'} J e_{vv'}\, s_v s_{v'}$$

$$\text{where } e_{vv'} = \begin{cases} +1 & E_{vv'} = I, \\ -1 & E_{vv'} = X. \end{cases}$$

Toric code and the random-bond Ising model

Step 0: Code and noise model

Toric code with iid bit-flips

$$\Pr(X_e) = p, \qquad \Pr(I_e) = 1 - p.$$

Step 1: Degrees of freedom

$s_v = \pm 1$ on each vertex v

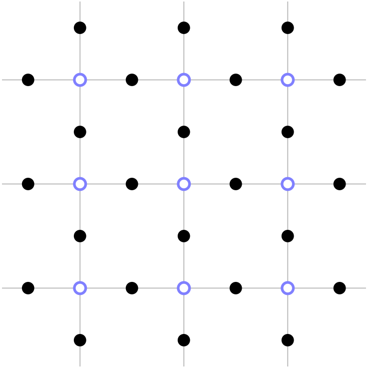
Step 2: Interactions

$$H_I = -\sum_{v \sim v'} J\, s_v s_{v'}$$

Step 3: Disorder

$$H_E = -\sum_{v \sim v'} J e_{vv'}\, s_v s_{v'}$$

$$\text{where } e_{vv'} = \begin{cases} +1 & E_{vv'} = I, \\ -1 & E_{vv'} = X. \end{cases}$$



Toric code and the random-bond Ising model

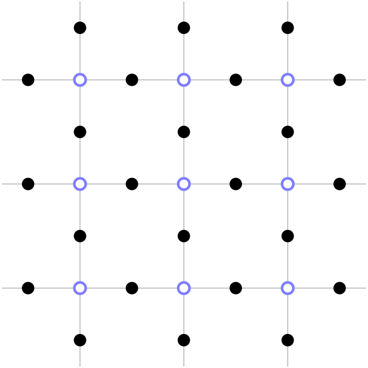
Step 0: Code and noise model

Toric code with iid bit-flips

$$\Pr(X_e) = p, \qquad \Pr(I_e) = 1 - p.$$

Step 1: Degrees of freedom

$s_v = \pm 1$ on each vertex v



Step 2: Interactions

$$H_I = -\sum_{v \sim v'} J\, s_v s_{v'}$$

Step 3: Disorder

$$H_E = -\sum_{v \sim v'} J e_{vv'}\, s_v s_{v'}$$

where $e_{vv'} = \begin{cases} +1 & E_{vv'} = I, \\ -1 & E_{vv'} = X. \end{cases}$

Toric code and the random-bond Ising model

Step 0: Code and noise model

Toric code with iid bit-flips

$$\Pr(X_e) = p, \qquad \Pr(I_e) = 1 - p.$$

Step 1: Degrees of freedom

$s_v = \pm 1$ on each vertex v

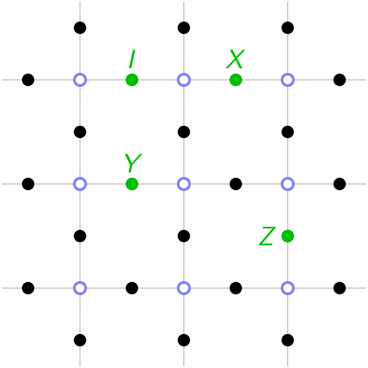
Step 2: Interactions

$$H_I = -\sum_{v\sim v'} J\, s_v s_{v'}$$

Step 3: Disorder

$$H_E = -\sum_{v\sim v'} J e_{vv'}\, s_v s_{v'}$$

$$\text{where } e_{vv'} = \begin{cases} +1 & E_{vv'} = I, \\ -1 & E_{vv'} = X. \end{cases}$$



$$[\![\sigma_e, S_v]\!] = \begin{cases} -1 & \text{if } \sigma = Y, Z,\ v \in e, \\ +1 & \text{else.} \end{cases}$$

Toric code and the random-bond Ising model

Step 0: Code and noise model

Toric code with iid bit-flips

$$\Pr(X_e) = p, \qquad \Pr(I_e) = 1 - p.$$

Step 1: Degrees of freedom

$s_v = \pm 1$ on each vertex v

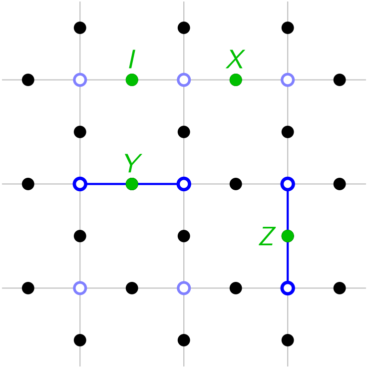
Step 2: Interactions

$$H_I = -\sum_{v \sim v'} J\, s_v s_{v'}$$

Step 3: Disorder

$$H_E = -\sum_{v \sim v'} J e_{vv'}\, s_v s_{v'}$$

$$\text{where } e_{vv'} = \begin{cases} +1 & E_{vv'} = I, \\ -1 & E_{vv'} = X. \end{cases}$$



$$[\![\sigma_e, S_v]\!] = \begin{cases} -1 & \text{if } \sigma = Y, Z,\ v \in e, \\ +1 & \text{else.} \end{cases}$$

Toric code and the random-bond Ising model

Step 0: Code and noise model

Toric code with iid bit-flips

$\Pr(X_e) = p, \quad \Pr(I_e) = 1 - p.$

Step 1: Degrees of freedom

$s_v = \pm 1$ on each vertex v

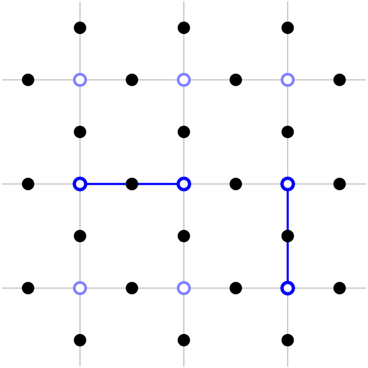
Step 2: Interactions

$$H_I = -\sum_{v \sim v'} J\, s_v s_{v'}$$

Step 3: Disorder

$$H_E = -\sum_{v \sim v'} J e_{vv'}\, s_v s_{v'}$$

where $e_{vv'} = \begin{cases} +1 & E_{vv'} = I, \\ -1 & E_{vv'} = X. \end{cases}$



Toric code and the random-bond Ising model

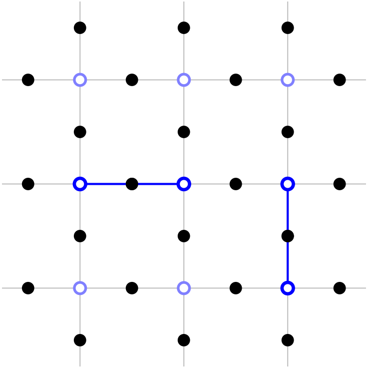
Step 0: Code and noise model

Toric code with iid bit-flips

$\Pr(X_e) = p, \quad \Pr(I_e) = 1 - p.$

Step 1: Degrees of freedom

$s_v = \pm 1$ on each vertex v



Step 2: Interactions

$$H_I = -\sum_{v\sim v'} J\, s_v s_{v'}$$

Step 3: Disorder

$$H_E = -\sum_{v\sim v'} J e_{vv'}\, s_v s_{v'}$$

where $e_{vv'} = \begin{cases} +1 & E_{vv'} = I, \\ -1 & E_{vv'} = X. \end{cases}$

$[\![\sigma, S_k]\!] \;\to\; [\![\sigma, ES_k]\!]$

Toric code and the random-bond Ising model

Step 0: Code and noise model

Toric code with iid bit-flips $\quad \mathrm{Pr}(X_e) = p, \quad \mathrm{Pr}(I_e) = 1 - p.$

Step 1: Degrees of freedom

$s_v = \pm 1$ on each vertex v

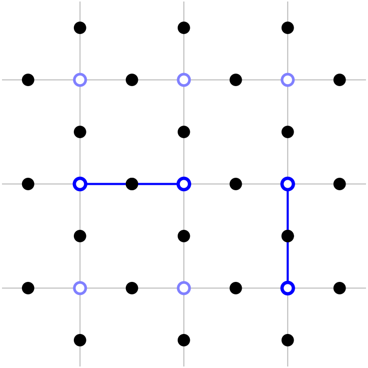
Step 2: Interactions

$$H_I = -\sum_{v \sim v'} J \, s_v s_{v'}$$

Step 3: Disorder

$$H_E = -\sum_{v \sim v'} J e_{vv'} \, s_v s_{v'}$$

where $e_{vv'} = \begin{cases} +1 & E_{vv'} = I, \\ -1 & E_{vv'} = X. \end{cases}$



$[\![\sigma, S_k]\!] \;\rightarrow\; [\![\sigma, ES_k]\!]$

Toric code and the random-bond Ising model

Step 0: Code and noise model

Toric code with iid bit-flips

$\Pr(+J) = p, \quad \Pr(-J) = 1 - p.$

Step 1: Degrees of freedom

$s_v = \pm 1$ on each vertex v

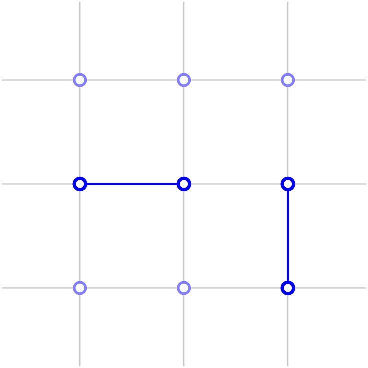
Step 2: Interactions

$$H_I = -\sum_{v \sim v'} J \, s_v s_{v'}$$

Step 3: Disorder

$$H_E = -\sum_{v \sim v'} J e_{vv'} \, s_v s_{v'}$$

$$\text{where } e_{vv'} = \begin{cases} +1 & E_{vv'} = I, \\ -1 & E_{vv'} = X. \end{cases}$$

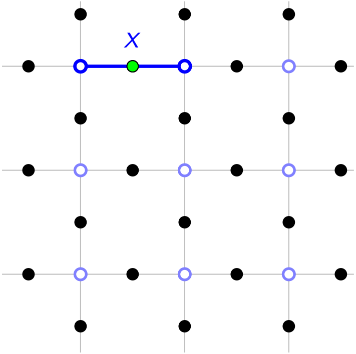


$\pm J$ Random-bond Ising Model

Other independent examples

Toric code

Bit-flip → Random-bond Ising[1]
Indep. $X\&Z$ → 2×Random-bond Ising
Depolarising → Random 8-vertex model[2]



Colour code

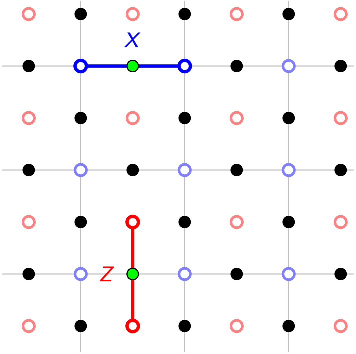
Bit-flip → Random 3-spin Ising
Indep. $X\&Z$ → 2×Random 3-spin Ising
Depolarising → Random interacting 8-vertex[2]

[1] Dennis et.al., JMP 2002, `doi:10/cs2mtf`, `arXiv:quant-ph/0110143`
[2] Bombin et.al., PRX 2012, `doi:10/crz5`, `arXiv:1202.1852`

Other independent examples

Toric code

Bit-flip → Random-bond Ising[1]
Indep. $X\&Z$ → 2×Random-bond Ising
Depolarising → Random 8-vertex model[2]



Colour code

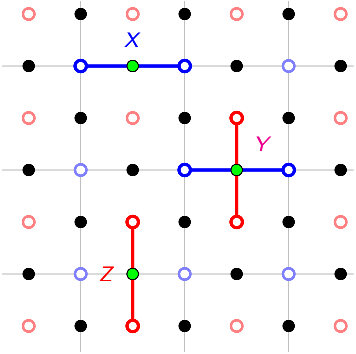
Bit-flip → Random 3-spin Ising
Indep. $X\&Z$ → 2×Random 3-spin Ising
Depolarising → Random interacting 8-vertex[2]

[1] Dennis et.al., JMP 2002, `doi:10/cs2mtf`, `arXiv:quant-ph/0110143`
[2] Bombin et.al., PRX 2012, `doi:10/crz5`, `arXiv:1202.1852`

Other independent examples

Toric code

Bit-flip → Random-bond Ising[1]
Indep. $X\&Z$ → $2\times$Random-bond Ising
Depolarising → Random 8-vertex model[2]



Colour code

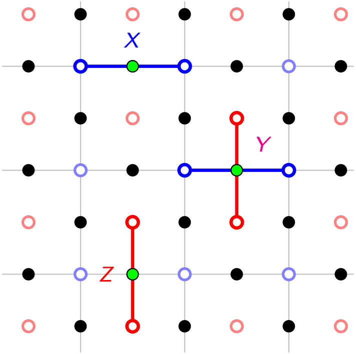
Bit-flip → Random 3-spin Ising
Indep. $X\&Z$ → $2\times$Random 3-spin Ising
Depolarising → Random interacting 8-vertex[2]

[1] Dennis et.al., JMP 2002, `doi:10/cs2mtf`, `arXiv:quant-ph/0110143`
[2] Bombin et.al., PRX 2012, `doi:10/crz5`, `arXiv:1202.1852`

Other independent examples

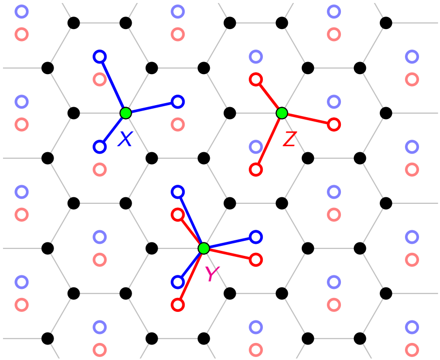
Toric code

Bit-flip → Random-bond Ising[1]
Indep. $X\&Z$ → 2×Random-bond Ising
Depolarising → Random 8-vertex model[2]



Colour code

Bit-flip → Random 3-spin Ising
Indep. $X\&Z$→ 2×Random 3-spin Ising
Depolarising → Random interacting 8-vertex[2]



[1] Dennis et.al., JMP 2002, `doi:10/cs2mtf`, `arXiv:quant-ph/0110143`
[2] Bombin et.al., PRX 2012, `doi:10/crz5`, `arXiv:1202.1852`

Error correction threshold as a quenched phase transition

Consider the free energy cost of a logical error L,

$$\Delta_E(L) = -\frac{1}{\beta}\log Z_{EL} + \frac{1}{\beta}\log Z_E.$$

Along the Nishimori line

$$\Delta_E(L) = \frac{1}{\beta}\log\frac{\Pr(\overline{E})}{\Pr(\overline{EL})},$$

which implies

$$\begin{aligned} \text{Below threshold :} \quad & \Delta_E(L) \to \infty \ \text{(in mean)} \\ \text{Above threshold :} \quad & \Delta_E(L) \to 0 \quad \text{(in prob.)} \end{aligned}$$

Error correction threshold as a quenched phase transition

Consider the free energy cost of a logical error L,

$$\Delta_E(L) = -\frac{1}{\beta}\log Z_{EL} + \frac{1}{\beta}\log Z_E .$$

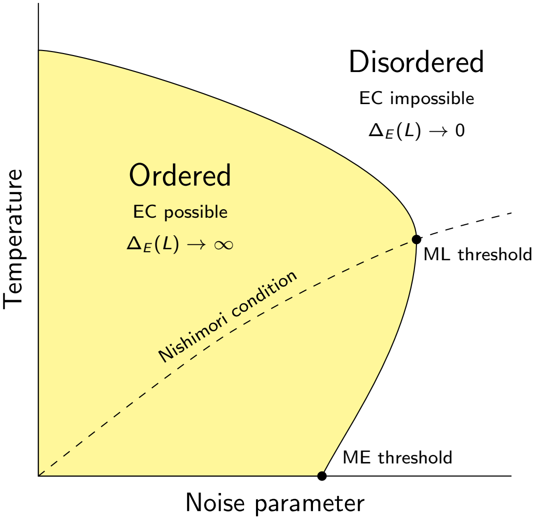
Along the Nishimori line

$$\Delta_E(L) = \frac{1}{\beta}\log\frac{\Pr(\overline{E})}{\Pr(\overline{EL})},$$

which implies

$$\begin{aligned}\text{Below threshold :} \qquad & \Delta_E(L) \to \infty \text{ (in mean)}\\ \text{Above threshold :} \qquad & \Delta_E(L) \to 0 \quad \text{(in prob.)}\end{aligned}$$

Phase diagram sketch



Correlated case

The key point independence gave us was the ability to factor our noise model

$$\Pr(E) = \prod_i p_i(E_i).$$

We can generalise this to correlated models:

Factored distribution

An error model factors over regions $\{R_j\}_j$ if there exist $\phi_j : \mathcal{P}_{R_j} \to \mathbb{R}$ such that

$$\Pr(E) = \prod_j \phi_j\left(E_{R_j}\right)$$

This model includes many probabilistic graphical models, such as Bayesian Networks and Markov/Gibbs Random Fields.

Correlated case

The key point independence gave us was the ability to factor our noise model

$$\Pr(E) = \prod_i p_i(E_i).$$

We can generalise this to correlated models:

Factored distribution

An error model factors over regions $\{R_j\}_j$ if there exist $\phi_j : \mathcal{P}_{R_j} \to \mathbb{R}$ such that

$$\Pr(E) = \prod_j \phi_j\left(E_{R_j}\right)$$

This model includes many probabilistic graphical models, such as Bayesian Networks and Markov/Gibbs Random Fields.

Correlated case

By construction, we can extend to the correlated case by changing $\sigma \in \mathcal{P}_i$ to $\sigma \in \mathcal{P}_{R_j}$:

$$H_E(\vec{s}) := -\sum_j \sum_{\sigma \in \mathcal{P}_{R_j}} J_j(\sigma) \llbracket \sigma, E \rrbracket \prod_{k : \llbracket \sigma, S_k \rrbracket = -1} s_k$$

Nishimori condition: $$\beta J_j(\sigma) = \frac{1}{|\mathcal{P}_{R_j}|} \sum_{\tau \in \mathcal{P}_{R_j}} \log \phi_j(\tau) \llbracket \sigma, \tau \rrbracket ,$$

As before we get that $Z_E = \Pr(\overline{E})$, and so the threshold manifests as a phase transition.

Correlated case

By construction, we can extend to the correlated case by changing $\sigma \in \mathcal{P}_i$ to $\sigma \in \mathcal{P}_{R_j}$:

$$H_E(\vec{s}) := -\sum_j \sum_{\sigma \in \mathcal{P}_{R_j}} J_j(\sigma) \llbracket \sigma, E \rrbracket \prod_{k:\llbracket \sigma, S_k \rrbracket = -1} s_k$$

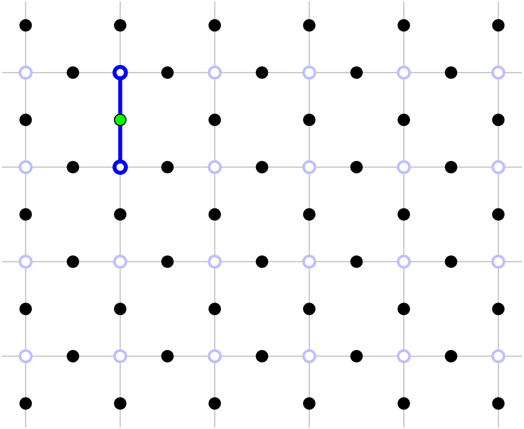
$$\text{Nishimori condition:} \qquad \beta J_j(\sigma) = \frac{1}{|\mathcal{P}_{R_j}|} \sum_{\tau \in \mathcal{P}_{R_j}} \log \phi_j(\tau) \llbracket \sigma, \tau \rrbracket \,,$$

As before we get that $Z_E = \Pr(\overline{E})$, and so the threshold manifests as a phase transition.

Correlated example

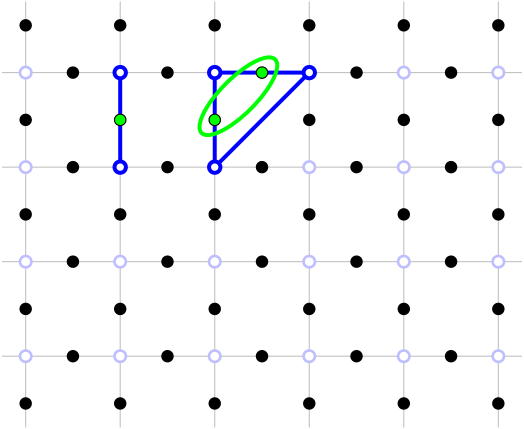
Toric code with correlated bit-flips

Correlations induce longer-range interactions



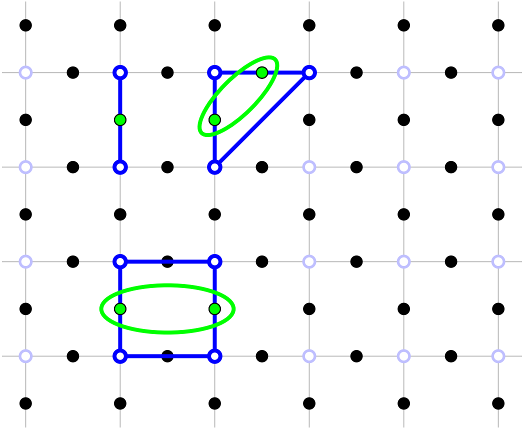
Correlated example

Toric code with correlated bit-flips
Correlations induce longer-range interactions



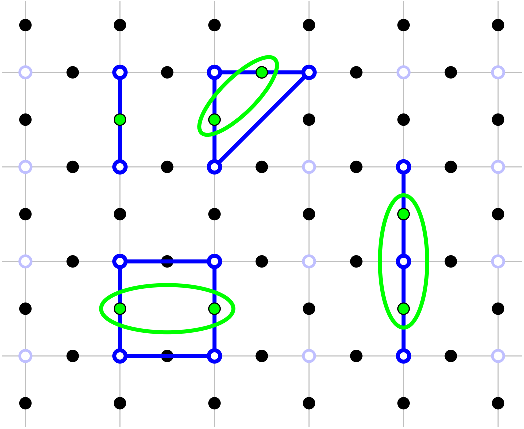
Correlated example

Toric code with correlated bit-flips
Correlations induce longer-range interactions

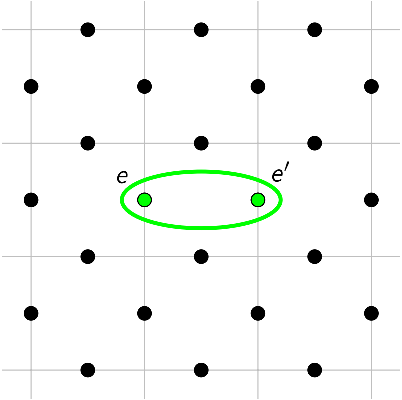


Correlated example

Toric code with correlated bit-flips
Correlations induce longer-range interactions



'Across plaquette' correlated bit-flips



This error model is entirely specified by the conditional error probabilities

$$
\begin{array}{ll}
\Pr(I_e|I_{e'}) & \Pr(I_e|X_{e'}) \\
\Pr(X_e|I_{e'}) & \Pr(X_e|X_{e'})
\end{array}
$$

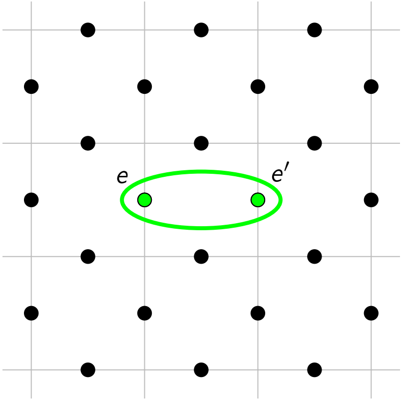
for all neighbouring edges e and e'.

For our purposes, it will convenient to parameterise things by

$$
p := \Pr(X_e), \qquad \eta := \frac{\Pr(X_e|X_{e'})}{\Pr(X_e|I_{e'})}.
$$

Here p is the marginal error rate, and η is a measure of the correlations.

‘Across plaquette’ correlated bit-flips



This error model is entirely specified by the conditional error probabilities

$$\Pr(I_e|I_{e'}) \qquad \Pr(I_e|X_{e'})$$
$$\Pr(X_e|I_{e'}) \qquad \Pr(X_e|X_{e'})$$

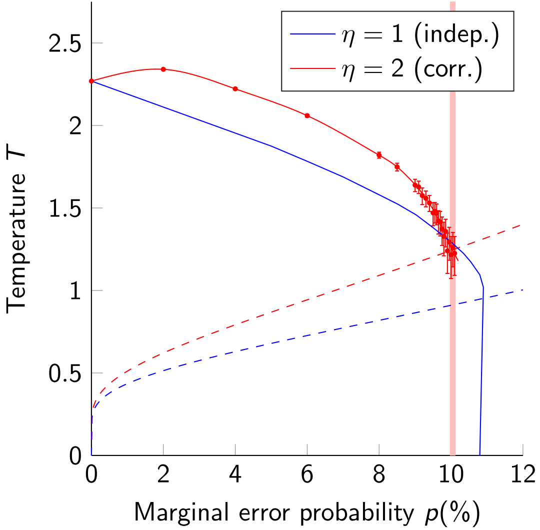
for all neighbouring edges e and e'.

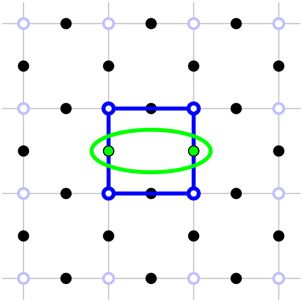
For our purposes, it will convenient to parameterise things by

$$p := \Pr(X_e), \qquad \eta := \frac{\Pr(X_e|X_{e'})}{\Pr(X_e|I_{e'})}.$$

Here p is the marginal error rate, and η is a measure of the correlations.

Monte Carlo simulations





Thresholds

Indep.: $p_{\mathrm{th}} = 10.917(3)\%$[1,2]
Corr.: $p_{\mathrm{th}} = 10.04(6)\%$

---

[1] Dennis et.al., JMP 2002, `doi:10/cs2mtf`, `arXiv:quant-ph/0110143`
[2] Toldin et.al., JSP 2009, `doi:10/c3r2kc`, `arXiv:0811.2101`

Decoding from partition functions

Along the Nishimori line, the maximum likelihood condition corresponds to maximising the partition function

$$\ell = \arg\max_{\ell} Z_{EL_\ell}.$$

Approximating Z_{EL_l} therefore allows us to approximate the ML decoder.

- Step 1: Measure the syndrome s
- Step 2: Construct an arbitrary error C_s which has syndrome s
- Step 3: Approximate $Z_{C_s L_l} = \Pr(\overline{C_s L_l})$ for each logical class l
- Step 4: Find the l such that $Z_{C_s L_l}$ is maximised
- Step 5: Apply $(C_s L_l)^{-1}$

Decoding from partition functions

Along the Nishimori line, the maximum likelihood condition corresponds to maximising the partition function

$$\ell = \arg\max_{\ell} Z_{EL_\ell}.$$

Approximating Z_{EL_l} therefore allows us to approximate the ML decoder.

- Step 1: Measure the syndrome s
- Step 2: Construct an arbitrary error C_s which has syndrome s
- Step 3: Approximate $Z_{C_s L_l} = \Pr(\overline{C_s L_l})$ for each logical class l
- Step 4: Find the l such that $Z_{C_s L_l}$ is maximised
- Step 5: Apply $(C_s L_l)^{-1}$

Decoding from (approximate) tensor network contraction

Partition functions can be expressed as tensor networks[1,2], allowing us to use approximate tensor network contraction schemes.

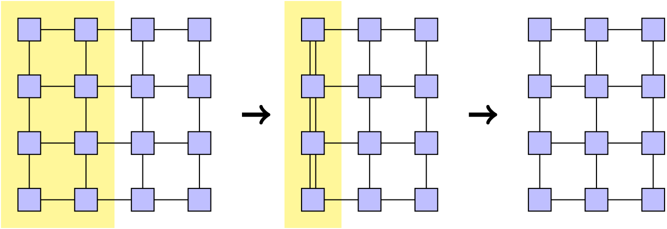
For 2D codes and locally correlated noise, this tensor network is also 2D. Here we can use the MPS-MPO approximation contraction scheme considered by Bravyi, Suchara and Vargo[3]:

[1] Verstraete et. al., PRL 2006, doi:10/dfgcz8, arXiv:quant-ph/0601075
[2] Bridgeman and Chubb, JPA 2017, doi:10/cv7m, arXiv:1603.03039
[3] Bravyi, Scuhara, Vargo, PRA 2014, doi:10/cv7n, arXiv:1405.4883

Decoding from (approximate) tensor network contraction

Partition functions can be expressed as tensor networks[1,2], allowing us to use approximate tensor network contraction schemes.

For 2D codes and locally correlated noise, this tensor network is also 2D. Here we can use the MPS-MPO approximation contraction scheme considered by Bravyi, Suchara and Vargo[3]:

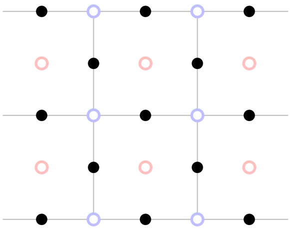


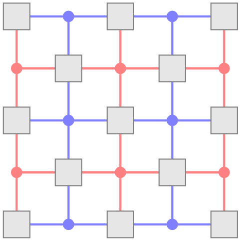
[1] Verstraete et. al., PRL 2006, doi:10/dfgcz8, arXiv:quant-ph/0601075
[2] Bridgeman and Chubb, JPA 2017, doi:10/cv7m, arXiv:1603.03039
[3] Bravyi, Scuhara, Vargo, PRA 2014, doi:10/cv7n, arXiv:1405.4883

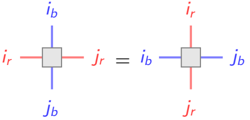
Decoding from (approximate) tensor network contraction

This gives an algorithm for (approximate) maximum likelihood decoding for any 2D code, subject to any locally correlated noise, generalising BSV.

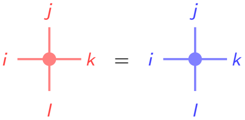
Indeed, applying this to iid noise in the surface code reproduces BSV:







$= p\left(E_e \cdot X^{i_b+j_b} Z^{i_r+j_r}\right)$



$= \delta_{i,j,k,l}$

Ongoing work: Surface codes on different graphs

The TN decoder lets us efficient probe the threshold of 2D topological codes.
What happens if we change the underlying graph?

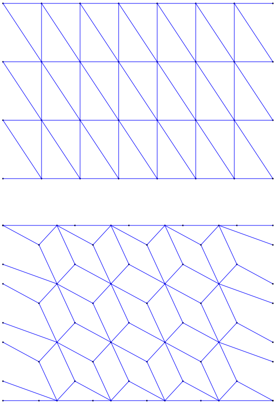
Raise or lower the connectivity

Irregular graphs

Ongoing work: Surface codes on different graphs

The TN decoder lets us efficient probe the threshold of 2D topological codes. What happens if we change the underlying graph?

Raise or lower the connectivity

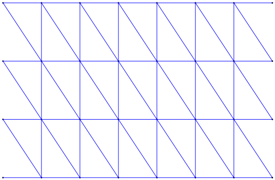


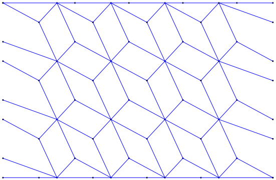
Irregular graphs

Ongoing work: Surface codes on different graphs

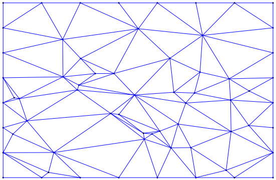
The TN decoder lets us efficient probe the threshold of 2D topological codes.
What happens if we change the underlying graph?

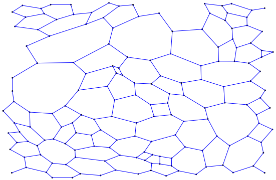
Raise or lower the connectivity





Irregular graphs





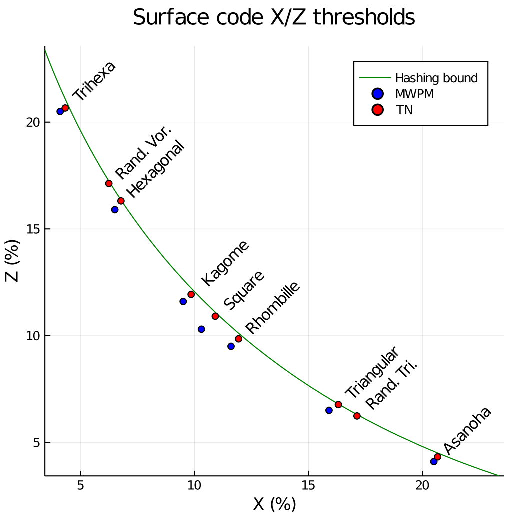
Ongoing work: Surface codes on different graphs

We find a trade-off between the X and Z thresholds.

Hashing bound: $h(p_x) + h(p_z) < 1$.

Pair matching studied earlier by Fujii et.al.[4]

We are currently running similar numerics for depolarising noise, and the colour code.



[4] Fujii et.al., `doi.org/d5sb`, `arXiv:1202.2743`

Further work

- TN decoding of LDPCs (ongoing work with Stefanos Kourtis)
- Use TN decoder to design codes for correlated noise

Thank you!

Stat Mech Mapping: arXiv:1809.10704, to appear in AIHPD
Tensor Network decoding: To appear arXiv:2009:?????

`me@christopherchubb.com`

`christopherchubb.com`

`@QuantumChubb`