Combining Hard and Soft Decoders for Hypergraph Product Codes

Antoine Grospellier[1] ; Lucien Grouès[1]; Anirudh Krishna[2]; Anthony Leverrier[1]

[1]INRIA Paris, [2]Université de Sherbrooke

July 31, 2019

Talk available at :
`https://www.youtube.com/watch?v=ZkfL59LGSc8`

Hypergraph product codes (Tillich, Zemor, 2009)

A powerful quantum code construction

- 2 classical codes $\rightarrow$ 1 CSS code
- repetition codes $\rightarrow$ toric code
- LDPC codes $\rightarrow$ LDPC code
- $[n, \Theta(n), \Theta(n)] \rightarrow [[N, \Theta(N), \Theta(\sqrt{N})]]$ with $N = \Theta(n^2)$
- expander codes $\rightarrow$ fault tolerance with **constant overhead** (Fawzi and al., arXiv:1808.03821)

Yet we don't know how to decode them well :

- **Small set flip decoder** : proved theoretically to decode quantum expander codes under **very low** error rates
- **Belief propagation decoder** : works very well in the classical case but **not** in the quantum case

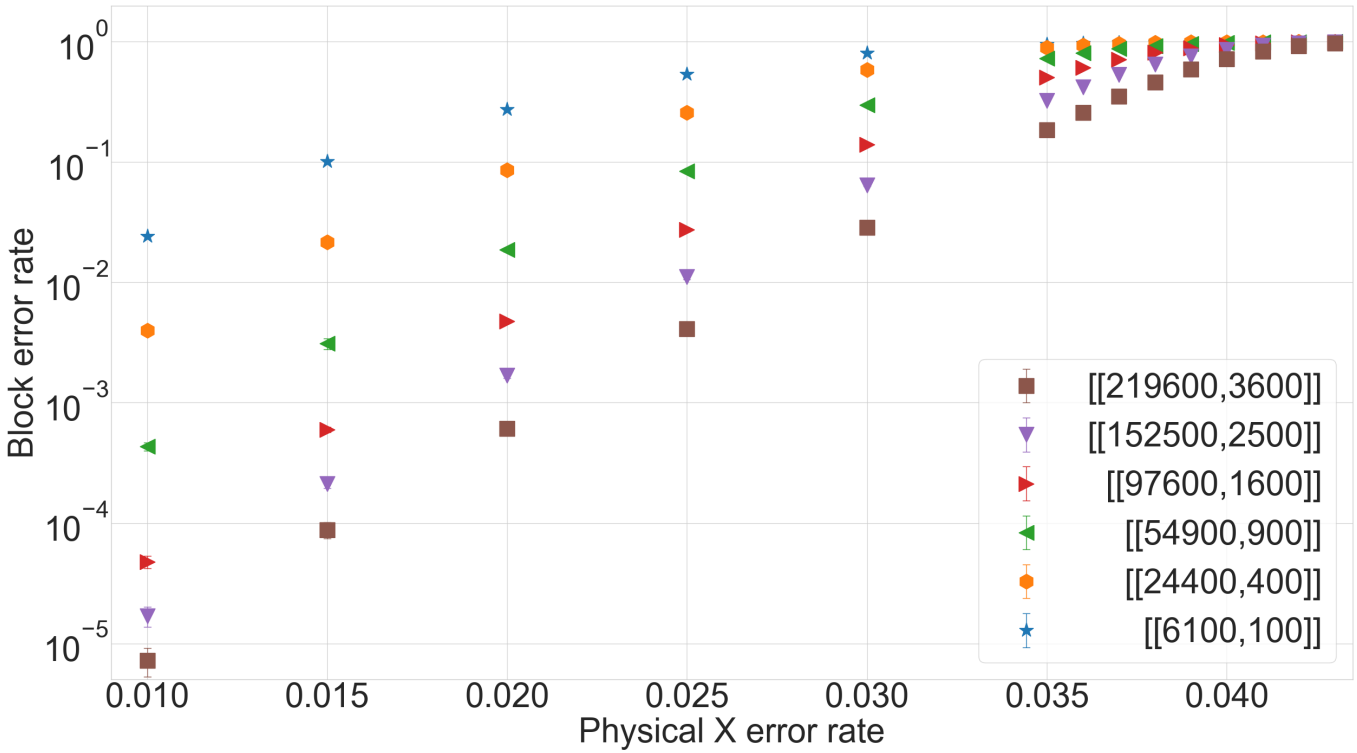
Our idea : combine both algorithms

Small set flip (SSF) : a hard decoder

Generalisation of the classical bit flip :
decreases the syndrome weight by flipping small sets of qubits

- With quantum expander codes $\rightarrow$ **constant overhead** fault tolerance
- Computation time $\rightarrow \Theta(N)$
- Theoretically $\rightarrow$ decodes under very low physical error rate
- In practice $\rightarrow$ decodes up to 4.6% on some LDPC hypergraph product codes (Grospellier, Krishna, arXiv:1810.03681)

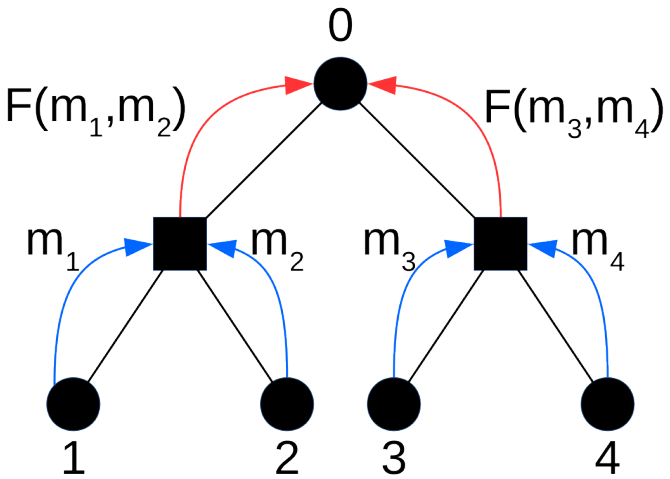
SSF: simulations (Grospellier, Krishna, arXiv:1810.03681)



- Product of a random 5,6-regular LDPC code with itself
- The threshold is around 4.6%

Belief propagation (BP) : a soft decoder

Computes for each bit $\mathbb{P}(\textit{faulty}|\textit{syndrome})$



- Based on the Tanner graph
- Message passing algorithm
- Number of rounds = browsing depth
- Exact on trees

Widely used in the classical case

- Tanner graph with large girth
  $\rightarrow$ good approximation
- computes all probabilities at once
  $\rightarrow$ very fast

But limited by quantum specifics (Poulin and al., arXiv:0801.1241)

- many cycles of length 4
  $\rightarrow$ girth too small
- code degeneracy
  $\rightarrow$ computes wrong probabilities

BP+SSF

Our contribution

Introducing BP+SSF :

- first decreases the size of the error with BP
- then corrects the residual error using SSF

Our simulations results

Independent X-Z noise ($p_x = p_z$)				
Code	**Rate**	**Stabilizers weight**	**Algorithm**	**Threshold**
Toric code	0%	4	MWPM	10.5%
4,5-hyperbolic surface code	10%	4 and 5	MWPM	2.5%
5,6 HGP code	1.6%	11	SSF	$\approx 4.6\%$
3,4 HGP code	4%	7	BP+SSF	$\approx 7.5\%$

[Kovalev and al., arXiv:1804.01950] Threshold around 7% on 3,4 HGP codes using estimated minimum weight decoder

[Panteleev and al., arXiv:1904.02703] Very good results on small cyclic HGP codes using BP+OSD (ordered statistical decoder)

Our simulations results

Independent X-Z noise ($p_x = p_z$)				
Code	**Rate**	**Stabilizers weight**	**Algorithm**	**Threshold**
Toric code	0%	4	MWPM	10.5%
4,5-hyperbolic surface code	10%	4 and 5	MWPM	2.5%
5,6 HGP code	1.6%	11	SSF	$\approx 4.6\%$
3,4 HGP code	4%	7	BP+SSF	$\approx 7.5\%$

Independent X-Z noise with syndrome errors ($p_x = p_z = p_{check}$)					
Code	**Rate**	**Stabilizers weight**	**Algorithm**	**Threshold**	**Single shot**
Toric code	0%	4	MWPM	2.9%	No
4,5-hyperbolic surface code	10%	4 and 5	MWPM	1.3%	No
3,4 HGP code	4%	7	$(\text{BP})^{\text{T}}$(BP+SSF)	$\approx 3\%$?	Yes

Table of content

Underlying classical code design

Several ways to build regular LDPC codes families :

- Random codes :
  - BP needs Tanner graphs with **few small cycles**
  - **Progressive edge growth algorithm** (PEG)
    $\rightarrow$ graphs with large girth very fast
  - Random local modifications + adapted scoring system
    $\rightarrow$ slower algorithm but **better results**
- BP + small **cyclic codes** hypergraph product promising (Panteleev and al., arXiv:1904.02703v1)
- **Quasi-cyclic codes** are more general

We will use **random 3,4-regular LDPC codes**

Plan

Error model and simulation protocol

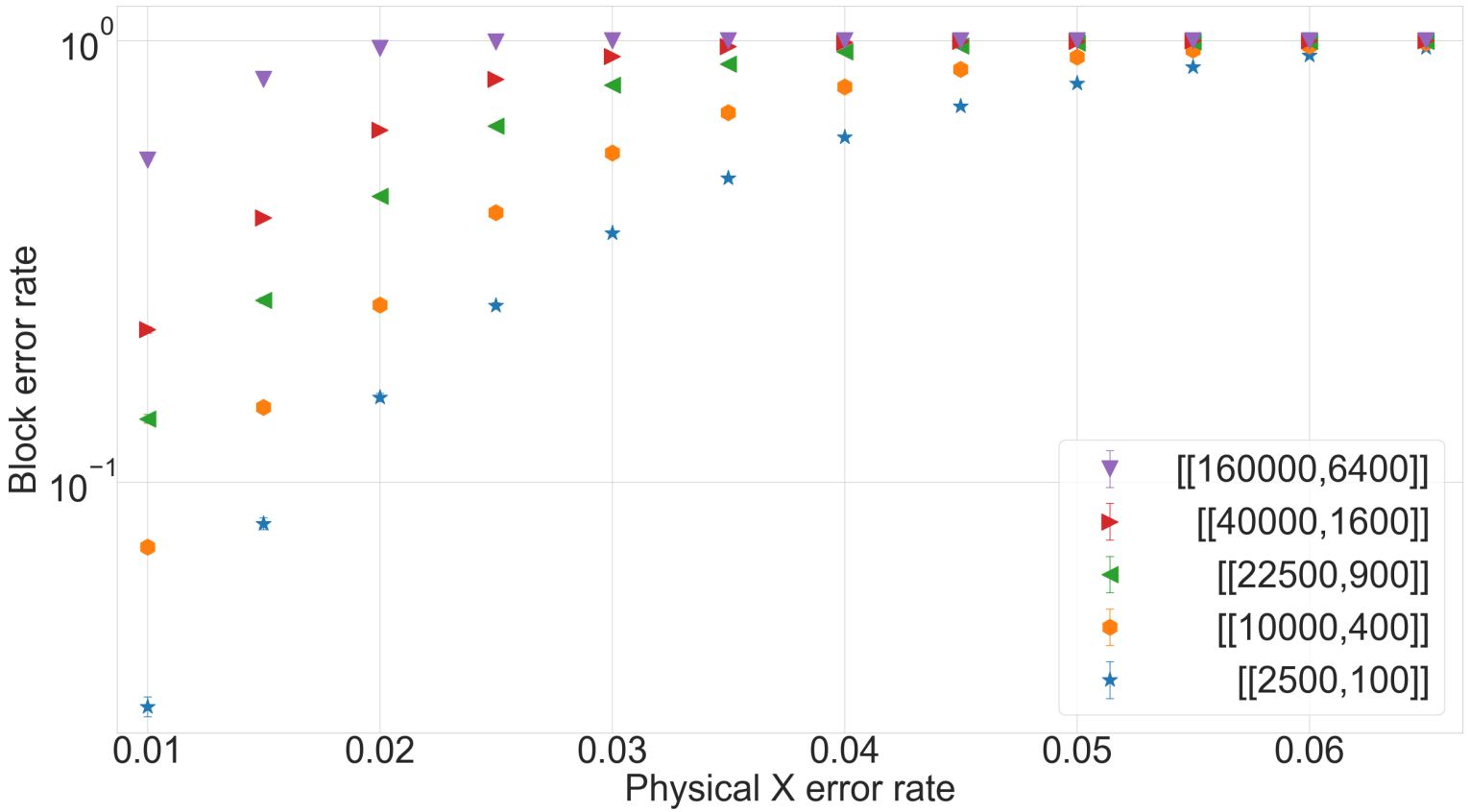
Error model : independent X-Z errors with $p_x = p_z = p$

- Errors corrected independently $\rightarrow$ graph girth length $= 8$
- Enough to simulate only X errors

Simulation protocol

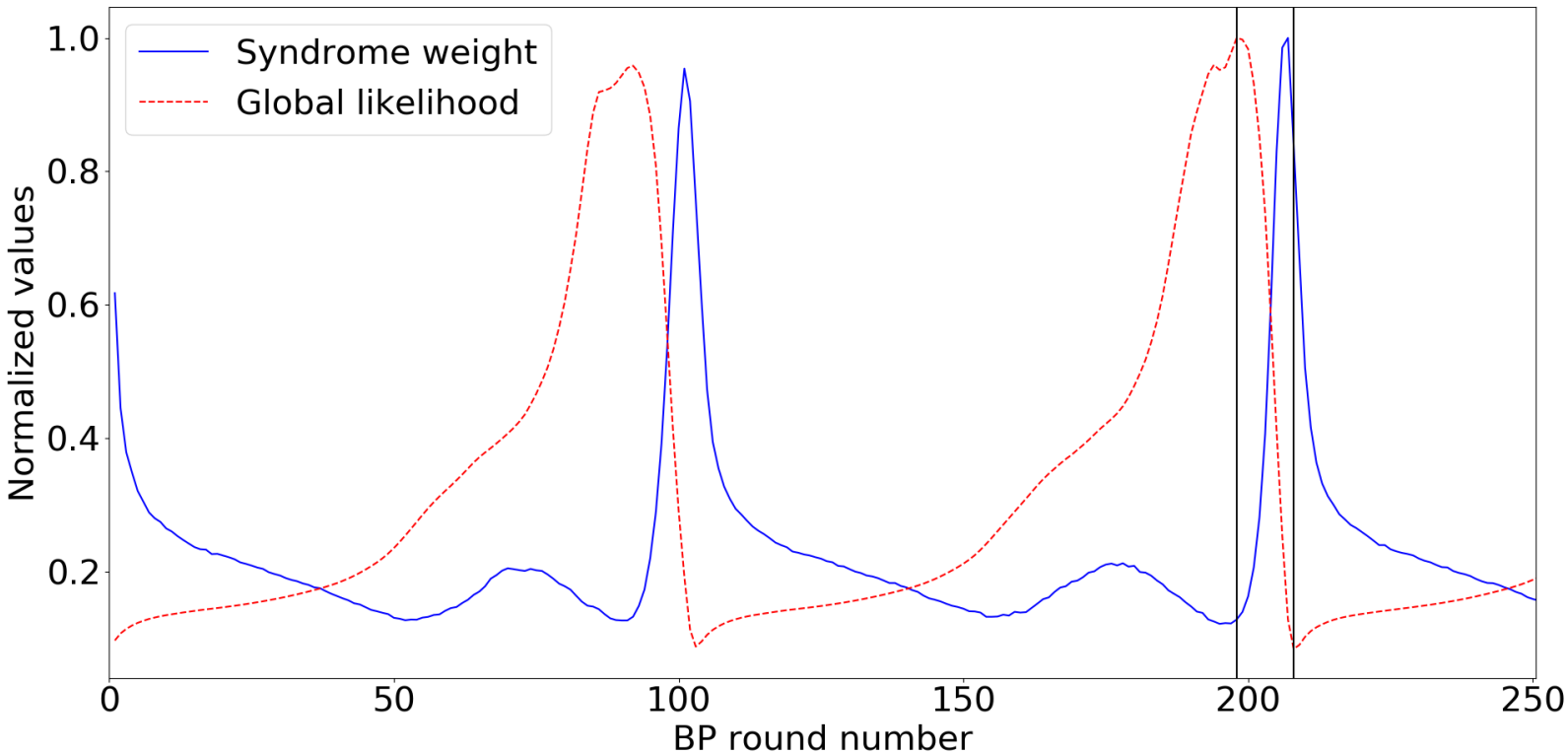
1. We flip each qubit with probability $p \rightarrow$ gives the error
2. We compute the syndrome
3. We run the decoder on the syndrome $\rightarrow$ gives the correction
4. (error $\Leftrightarrow$ correction) $\longrightarrow$ decoder succeeded

BP: simulations



- Performance worsen as we increase the code size
- No threshold found above 0.5%

Common BP failing behaviour



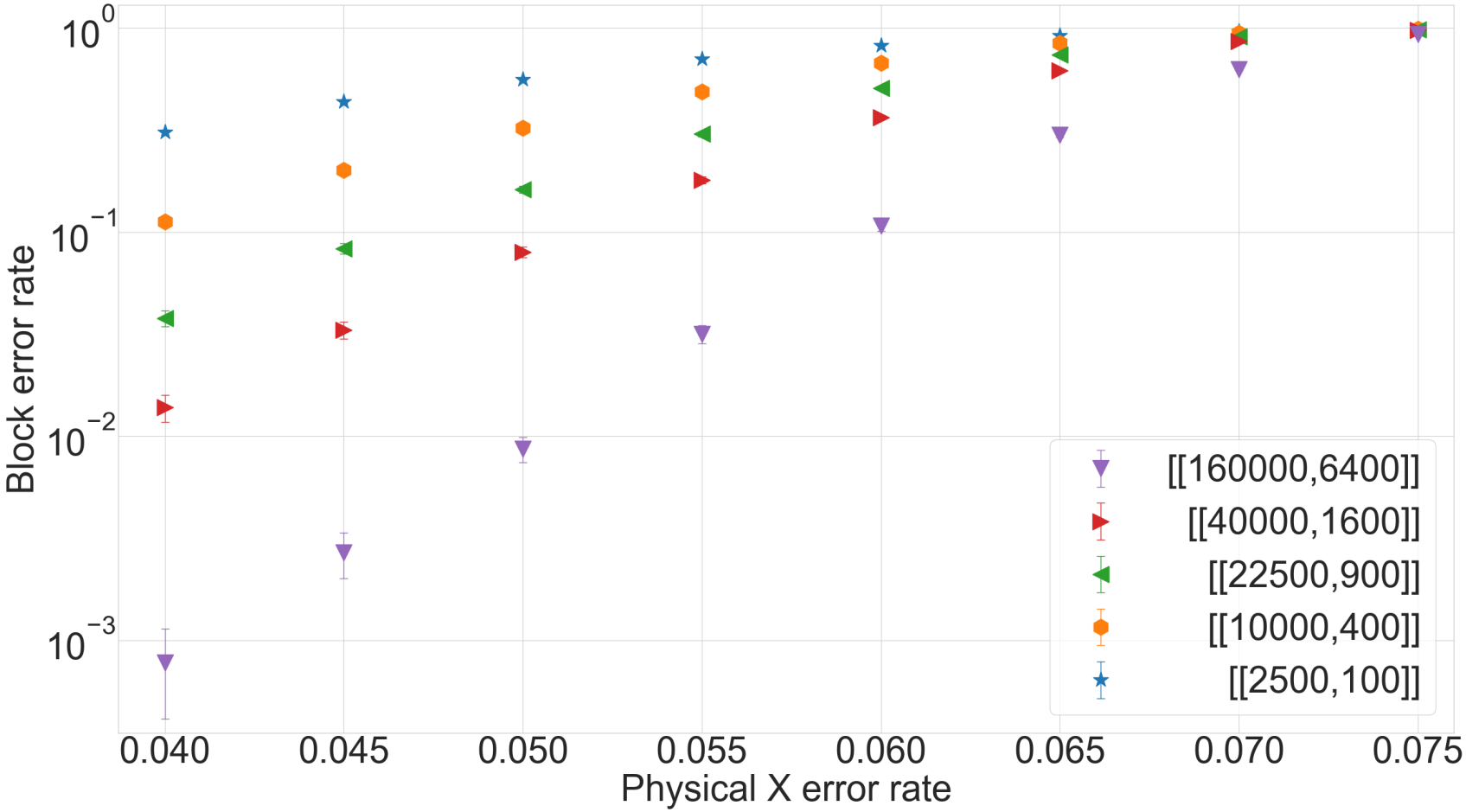
- BP doesn't converge $\rightarrow$ starts to oscillate
- maximum likelihood $\Leftrightarrow$ minimum syndrome weight

Combining BP with SSF

Almost no logical error → BP often can't reach a codeword

- Yet BP decreases a lot the syndrome weight
  → use SSF to close the gap
- Often high amplitude oscillations
  → we shouldn't stop BP at an arbitrary time
- Syndrome weight looks relevant
  → stop at its first local minimum
- Why the first minimum ?
  - easy to find and fast to compute
  - less rounds → smaller depth → **sees less cycles**

BP+SSF: simulation



- Threshold around 7.5% of physical errors probability
- Big improvement compared to the two algorithms alone

Improving the BP stopping condition

The stopping condition is vulnerable to bad first local minimum

How to solve it :

- Big fixed number of rounds but come back to the best state
- The best state can be the one with :
  - the smallest syndrome weight
  - the highest likelihood

Both tried with 100 rounds $\rightarrow$ overall unsatisfying results :

- improvement only for small codes
- curves having sweet values and crossing several times

Plan

From noiseless to noisy syndrome

Fault tolerance context → noisy syndrome measurement

- Are our algorithms still working in that case ?
- It might be needed to adapt them consequently
- We can't evaluate the performances the same way

Model used by Breuckmann and al. (arXiv:1703.00590):

- Several rounds of faulty syndrome and one last exact :
  - At the end of the calculus the information is classical and we can measure exactly the syndrome
- Unreliable syndrome measurement → most algorithms need to repeat the measurement many times at each round ($\sqrt{n}$ for the toric code)
- In our case only one measurement needed → **single-shot** property

How to evaluate the performance

Same method as Brown, Nickerson, Browne (arXiv:1503.08217) :

- The threshold depends on the number of faulty rounds (T)
- Under this limit : larger codes $\rightarrow$ information lasts longer
- The dependence on T is not trivial

  $\rightarrow$ we need to do a fitting of the curves (not done yet)

Simulations protocol

Algorithm: $(D_1)^T D_2$

input : A codeword

output: Whether we managed to conserve the codeword

for *i=1 to T* **do**

- Apply on each qubit X error with probability p
- Compute syndrome and flip each bit with probability p
- Run decoder D_1 on syndrome assuming the noise is still IID
- Apply correction

end

Apply on each qubit X error with probability p

Compute syndrome

Run decoder D_2 on syndrome assuming the noise is still IID

Apply correction

Succeeds if the new word is equivalent to the input codeword

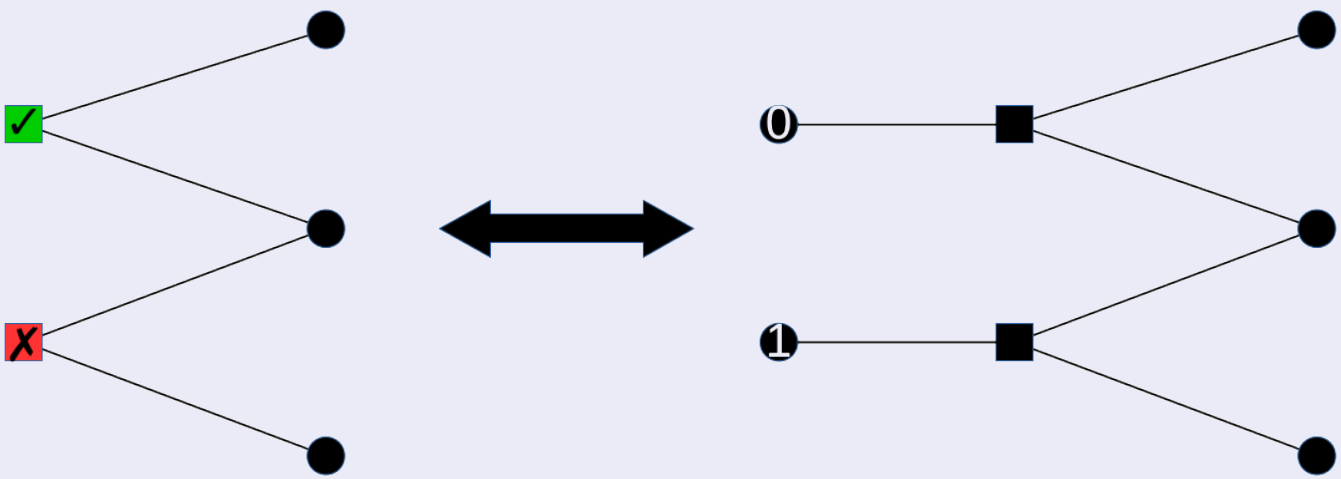
SSF and BP for the noisy syndrome case

Adaptation of the SSF

Acts as if the syndrome measurements were perfect :

- Tries to reduce the weight of the noisy syndrome
- Its guess on the syndrome error are the unsatisfied checks left

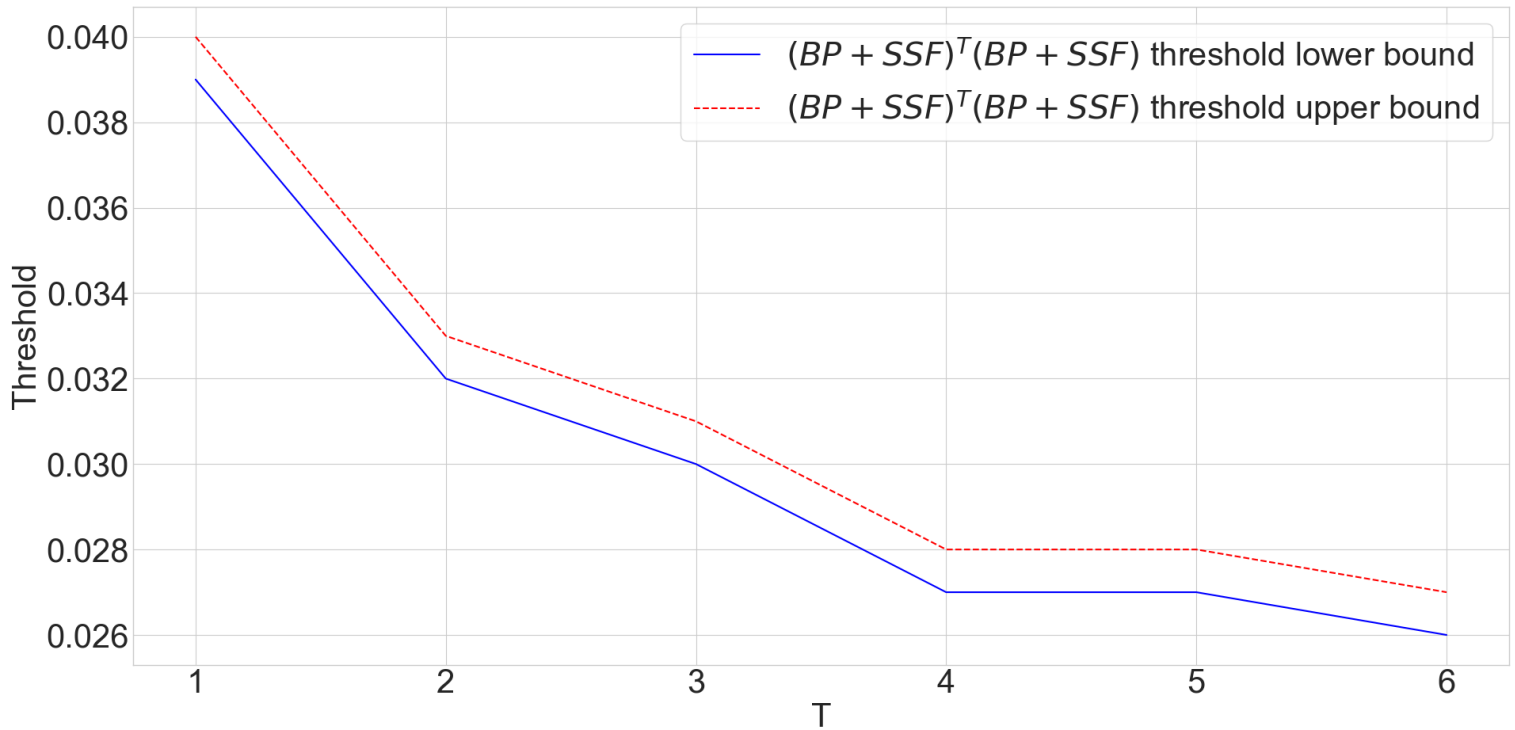
Adaptation of the BP



BP+SSF for the noisy syndrome case

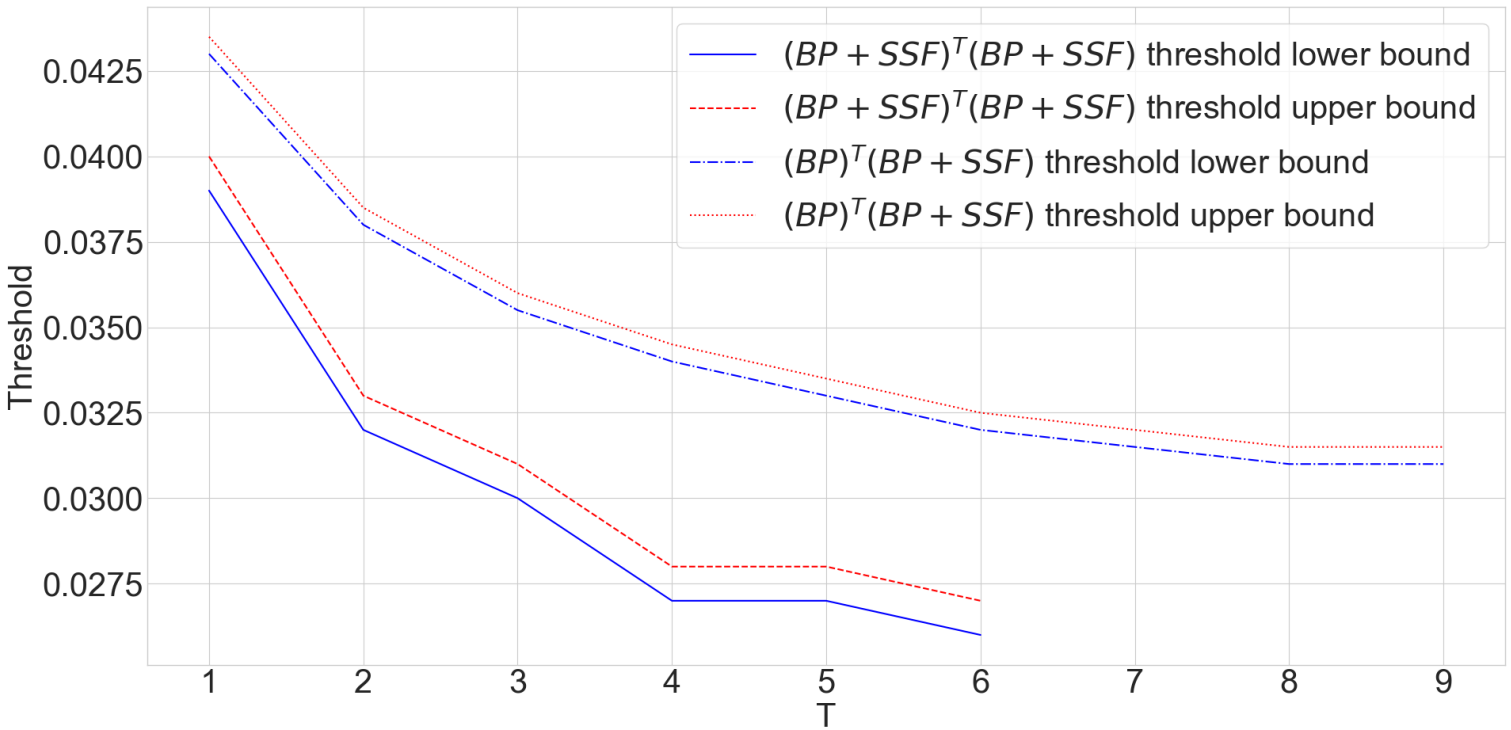
- BP computes the probability of the new bits to be equal to 1
  $\rightarrow$ this gives us a guess for the syndrome error
- BP applies both the qubits and syndrome correction
  $\rightarrow$ the SSF gets the *corrected* new syndrome as input
- **Contrary to SSF the syndrome error guess of BP isn't equal to the unsatisfied checks left**

$(BP+SSF)^T(BP+SSF)$



- Too early to conclude but promising
- The limit looks above 2.5%

$(BP)^T(BP+SSF)$



- Too early to conclude but even better results
- The limit should be around 3%

$(BP)^T(BP+SSF) > (BP+SSF)^T(BP+SSF)$

Possible explanation

- BP isn't good at guessing the syndrome error :
  above 2.5% BP increases the weight of the syndrome error
- SSF doesn't take into account the syndrome errors
  $\rightarrow$ may take us very far from valid codewords
- BP takes these errors into account
  $\rightarrow$ keeps us in the good area
- SSF still needed once the syndrome measurement is exact
  $\rightarrow$ closes the gap

Gives **better** results with **faster** computation !

Plan

Summary

What we did :

Heuristic combining BP and SSF

- Noiseless syndrome case $\rightarrow$ far better than SSF : **better rate** ($1.6\% \rightarrow 4\%$) and **threshold** ($4.6\% \rightarrow 7.5\%$)
- Noisy syndrome case $\rightarrow$ started to get promising results : threshold $\approx 3\%$?

What's next?

- Try to improve our quantum codes
  $\rightarrow$ e.g. use 2 different classical codes to deal with depolarising channel
- better version of BP for quantum LDPC codes (taking degeneracy in consideration) $\rightarrow$ open question

Thank you