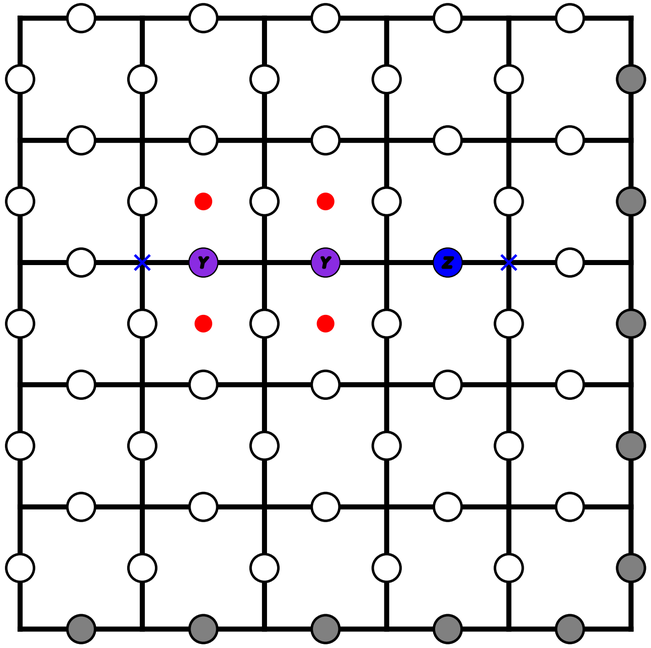
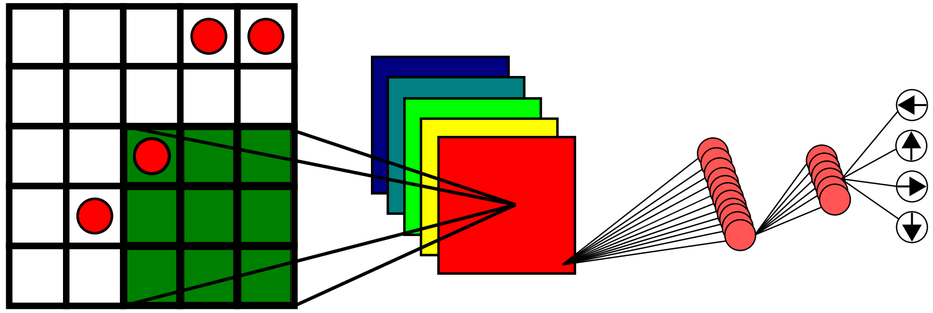
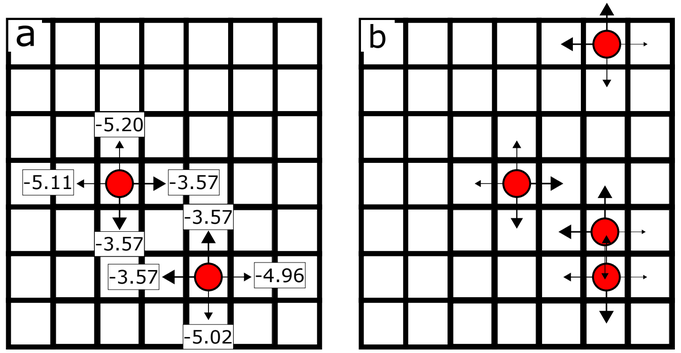
Quantum error correction for the toric code using deep reinforcement learning

Mats Granath
Department of Physics
University of Gothenburg

Machine Learning for Quantum Technology
Max Planck Institute for the Science of Light
May 9th, 2019







Philip Andreasson, Joel Johansson, Simon Liljestrand, MG, arXiv:1811.12338

Mattias Eliasson, David Fitzek, MG, in progress

Bottom line

We do the “simplest” error correction problem for a topological code

- Periodic boundary conditions
- No measurement noise/perfect syndrome
- only bit flip noise (initially, can also do depolarizing)

Still challenging for reinforcement learning: deep Q-networks needed

Allows for easy benchmark

The toric code

Fault-tolerant quantum computation by anyons

A.Yu. Kitaev*

L.D. Landau Institute for Theoretical Physics, 117940, Kosygina St. 2, Germany

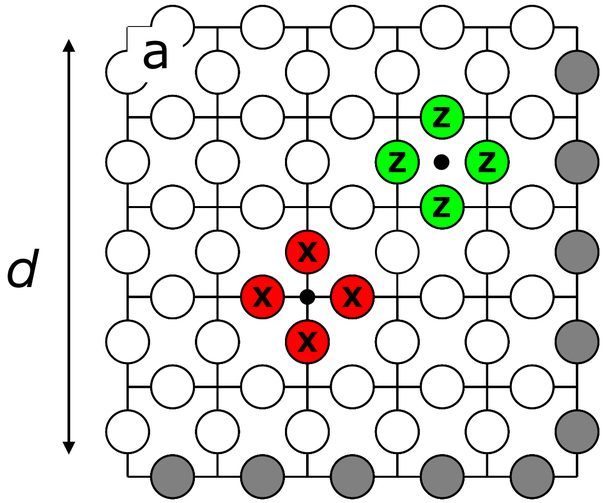
Received 20 May 2002

JOURNAL OF MATHEMATICAL PHYSICS VOLUME 43, NUMBER 9 SEPTEMBER 2002

Topological quantum memory[a)]

Eric Dennis[b)]
Princeton University, Princeton, New Jersey 08544

Alexei Kitaev,[c)] Andrew Landahl,[d)] and John Preskill[e)]
Institute for Quantum Information, California Institute of Technology, Pasadena, California 91125



$$H = -\sum_{\alpha} \hat{P}_\alpha - \sum_{\nu} \hat{V}_\nu$$

$$\hat{P}_\alpha = \prod_{i \in \alpha} \sigma_i^z$$

$$\hat{V}_\nu = \prod_{i \in \nu} \sigma_i^x$$

Plaquette and Vertex stabilizers (parity checks)

$2d^2$ physical qubits, $2d^2$-2 independent stabilizers

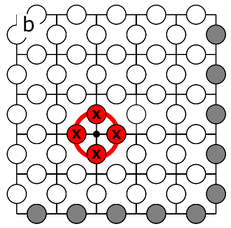
Ground state

consider:

$$| \uparrow\uparrow\uparrow \cdots \rangle$$

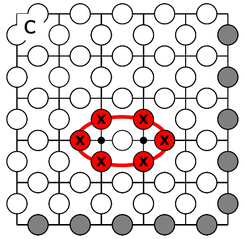
plaquette operator ground state:

act with vertex op:



still a plaquette ground state

act with two vertex op:



still a plaquette ground state

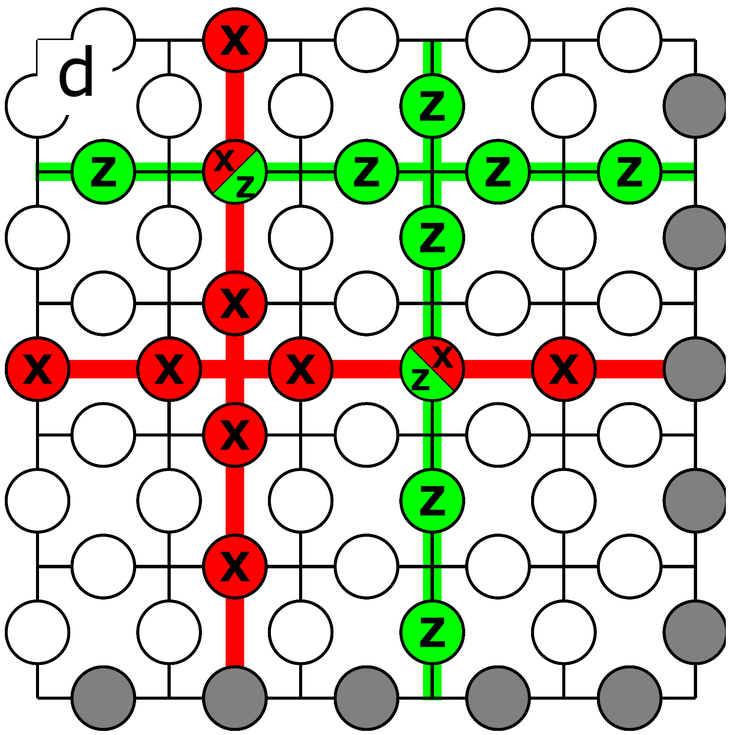
GS is symmetric superposition of all *trivial* loops:

$$|\mathrm{GS}_0\rangle = \sum_{i \in \text{all trivial loops}} \mathrm{loop}_i | \uparrow\uparrow\uparrow \cdots \rangle$$

highly entangled

Ground state degeneracy

Non-trivial loops (encircling torus) X_1, X_2
are not products of vertex operators.



Four ground states/The logical qubit

$$\{|\mathrm{GS}_0\rangle, X_1|\mathrm{GS}_0\rangle, X_2|\mathrm{GS}_0\rangle, X_2X_1|\mathrm{GS}_0\rangle\}$$

Distinguished by ±1 eigenvalues of Z_1 and Z_2.

Corresponding to $2(d^2-1)$ independent stabilizers on $2d^2$ physical qubits.

Topologically protected qubit

Non-trivial loops=Logical bit-flip operators
Requires at least *d* physical bit-flip errors
code distance *d*

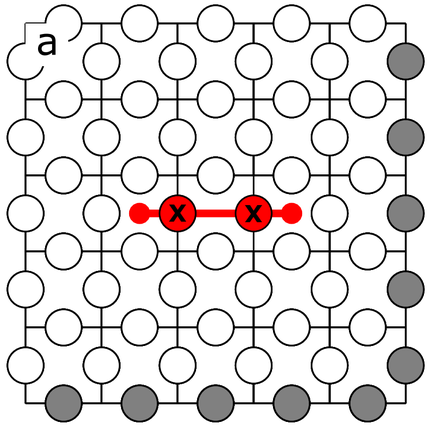
Error correction

consider bit-flip errors

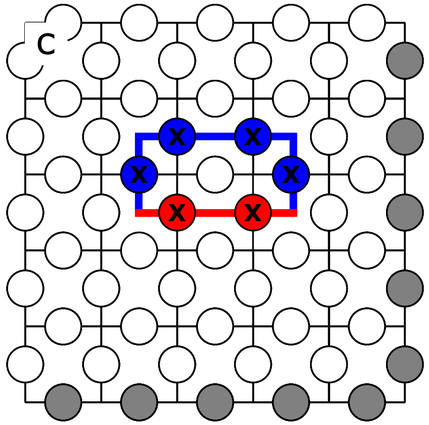
The *syndrome* (defects/bad plaquettes), is quantum non-demolition measurement

Ex.

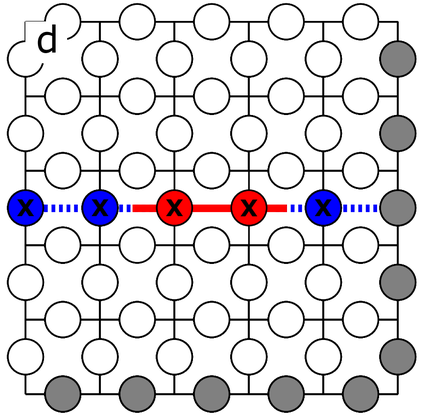
two neighbouring bit flip errors,
two defects



proper error correction
trivial loop



failed error correction
non-trivial loop



Standard algorithm to suggest error correcting strings:
Minimum Weight Perfect Matching (MWPM)/Blossom

J. Edmunds, 1965

Find shortest total correction string. (Which is the most likely)

Error models

Depolarizing

- **(1-p) no error**
- **p/3 X**
- **p/3 Y=XZ**
- **p/3 Z**

Uncorrelated

- **$(1-p)^2$ no error**
- **p(1-p) X**
- **p^2 Y=XZ**
- **p(1-p) Z**

Bit- and phase-flip errors
(i.e. plaquette and vertex errors).
are independent. Corrected separately.

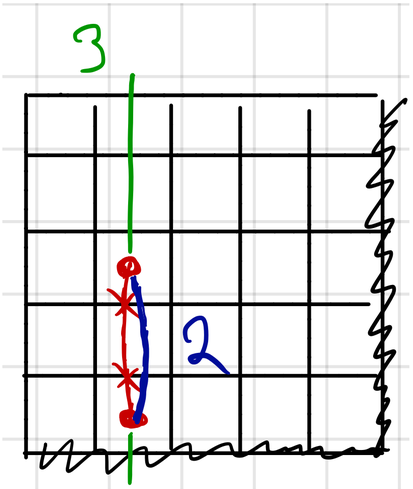
MWPM is (near) optimal

Minimum Weight Perfect Matching Low-p fail rate for bit-flip errors

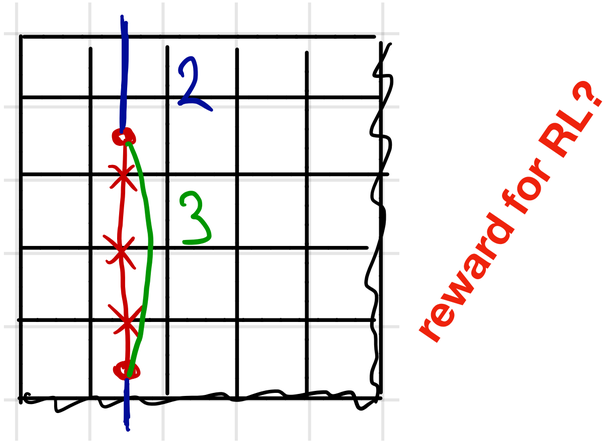
For p -> 0 we only need to consider error chains with minimal number of errors that can give failed error correction

Consider *d*=5:

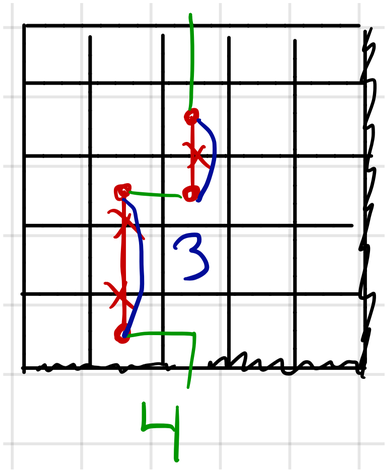
Two errors is always corrected successfully



Three errors in a row always gives failed error correction



Three errors not in a row always gives successful correction



MWPM asymptotic (lowest order in *p*) fail rate is:

$$p_L = 2d\binom{d}{\lceil d/2 \rceil} p^{\lceil d/2 \rceil}$$

Mats Granath, MLQT, Erlangen 2019

Deep reinforcement learning/Deep Q-learning

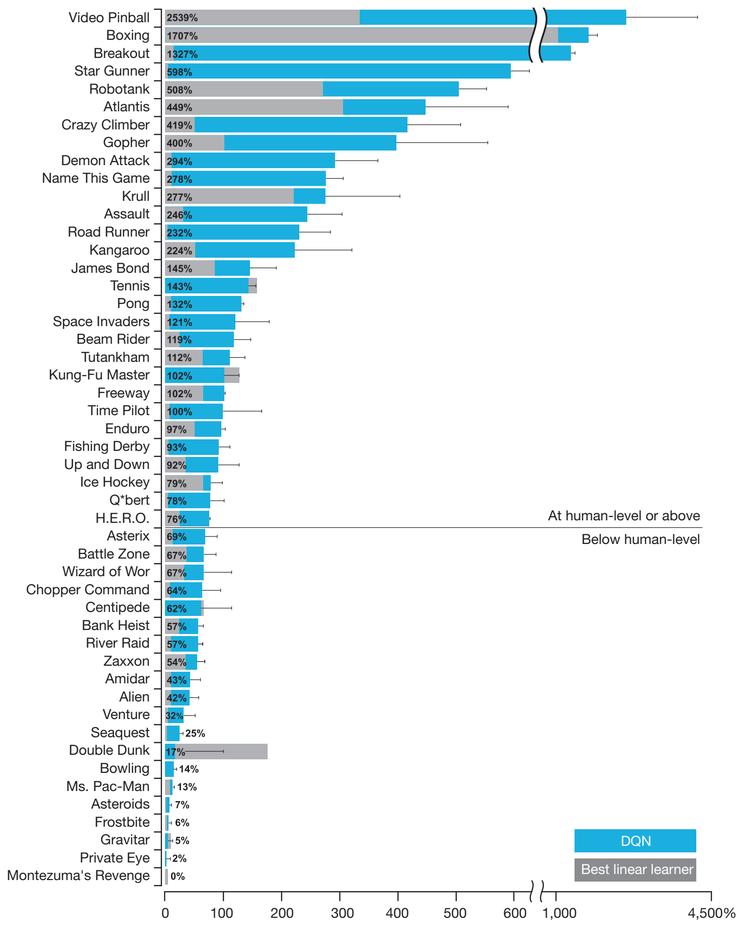
LETTER

2015

doi:10.1038/nature14236

Human-level control through deep reinforcement learning

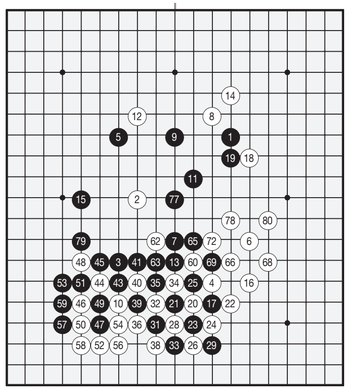
Volodymyr Mnih[1]*, Koray Kavukcuoglu[1]*, David Silver[1]*, Andrei A. Rusu[1], Joel Veness[1], Marc G. Bellemare[1], Alex Graves[1], Martin Riedmiller[1], Andreas K. Fidjeland[1], Georg Ostrovski[1], Stig Petersen[1], Charles Beattie[1], Amir Sadik[1], Ioannis Antonoglou[1], Helen King[1], Dharshan Kumaran[1], Daan Wierstra[1], Shane Legg[1] & Demis Hassabis[1]



2017

Mastering the game of Go without human knowledge

David Silver[1]*, Julian Schrittwieser[1]*, Karen Simonyan[1]*, Ioannis Antonoglou[1], Aja Huang[1], Arthur Guez[1], Thomas Hubert[1], Lucas Baker[1], Matthew Lai[1], Adrian Bolton[1], Yutian Chen[1], Timothy Lillicrap[1], Fan Hui[1], Laurent Sifre[1], George van den Driessche[1], Thore Graepel[1] & Demis Hassabis[1]



AlphaStar 2019

Mats Granath, MLQT, Erlangen 2019

Q-learning

- Agent in an environment described by a **state** *s*.
- Agent takes **actions** *a* to move between states, *s* -> *s*'.
- **Reward** (positive or negative) *r* is given depending on state/action.
- Agent learns **policy**, *π(s,a)*, to navigate environment for optimal accumulated reward (return) by exploring.

Q-function (action-value fcn) *Q(s,a)* **quantifies expected return from taking action** *a* **in state** *s* **and subsequently following the optimal policy.**

$$Q(s,a) = r + \gamma \max_{a'} Q(s',a')$$

γ<1 is discounting factor, better to get reward now than later

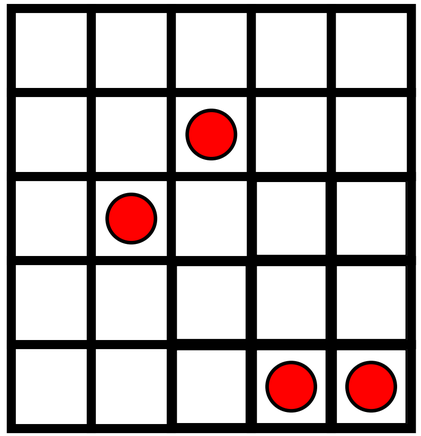
Explore to get reward and learn Q => optimal policy

Difficult if big world with many states and actions

Use Artificial Neural Network to represent Q-function
Deep Q-learning

Q-learning for the toric code

state is a syndrome
action is a bitflip=cardinal move of defect
reward, r=-1 per move (i.e. we aim to learn MWPM)



State space is very big
number of ways of placing N_S defects on d^2 sites:

$$\binom{d^2}{N_s} \approx \binom{49}{20} \sim 10^{13}$$

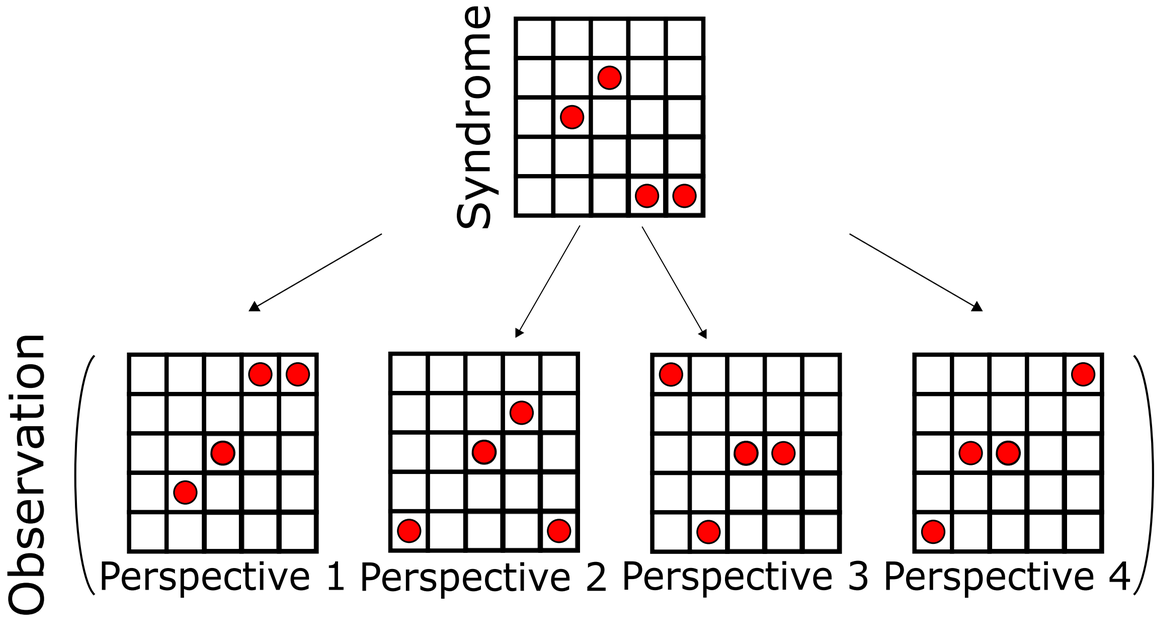
for d=7 and p=10%

Use deep Q-learning

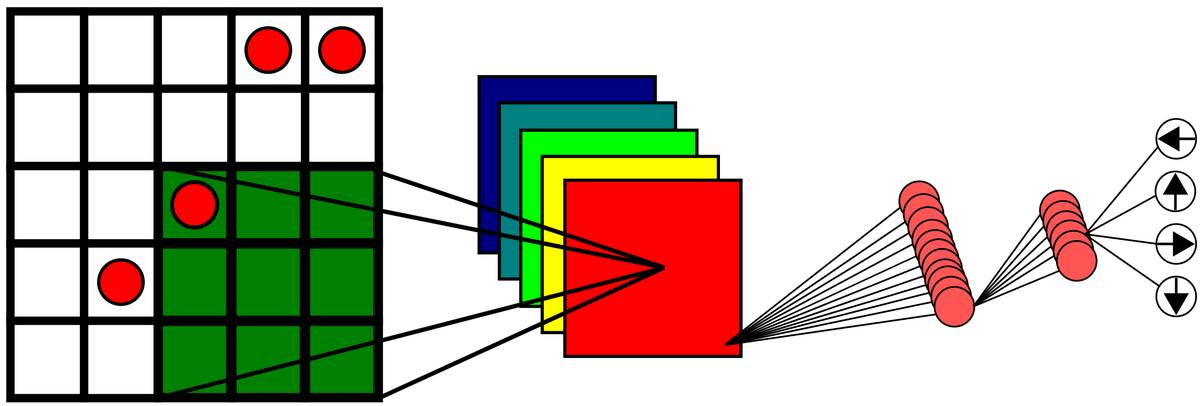
Mats Granath, MLQT, Erlangen 2019

Efficient implementation of Q-network

Use translational and rotational symmetry
to center each defect.



Convolutional NN



Mats Granath, MLQT, Erlangen 2019

Deep Q-network

Network gives Q-values for the 4 movements of the **central** defect.
Crucial simplification, fixed number (4) actions, and doesn't have to learn about boundaries.

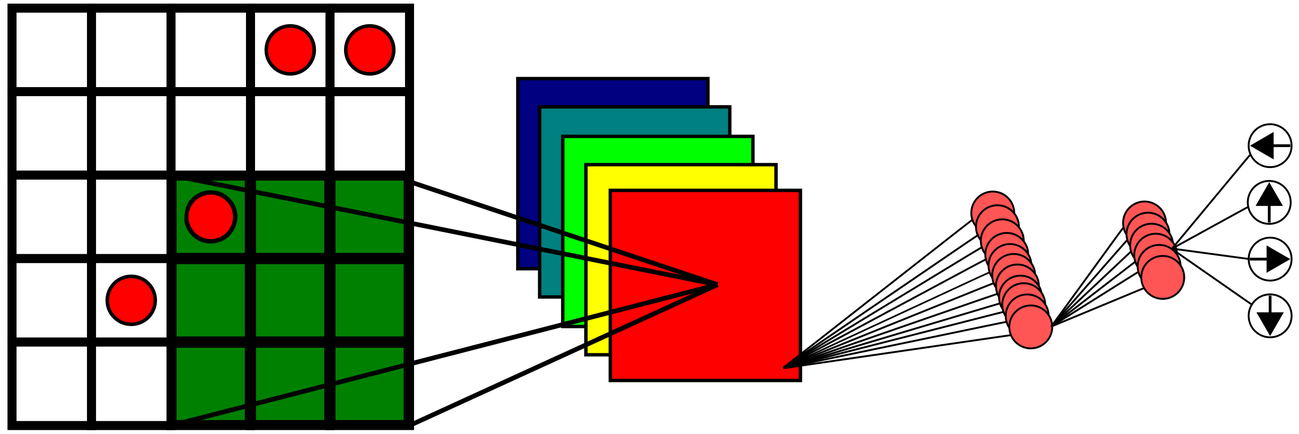


Table 2: Network architecture d=7.

#	Type	Size	# parameters
0	Input	7x7	
1	Conv.	512 filters; 3x3 size; 2-2 stride	5 120
2	FC	256 neurons	1 179 904
3	FC	128 neurons	32 896
4	FC	64 neurons	8 256
5	FC	32 neurons	2 080
6	FC (out)	4 neurons	132
			1 228 388

Experience replay is crucial for training

Significant reduction in number of parameters.
Size of state space for d=7, and N_S=20 defects (10% error)

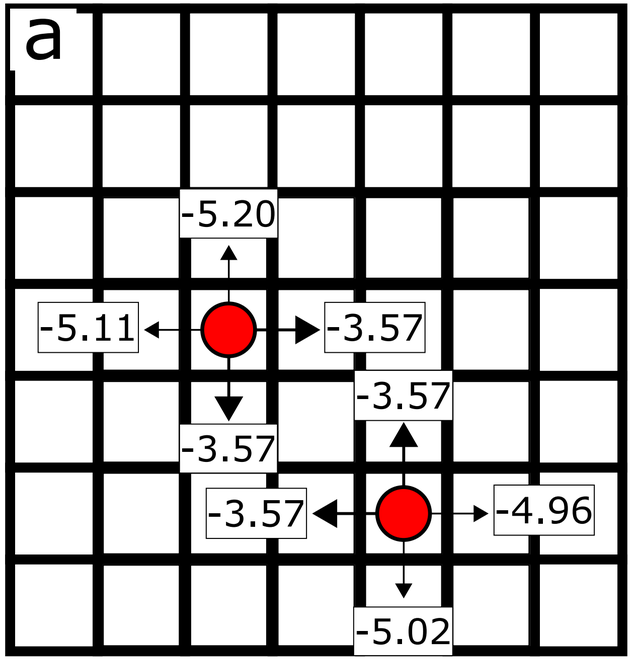
$$\binom{d^2}{N_s} \approx \binom{49}{20} \sim 10^{13}$$

Results. Converged Q-network.

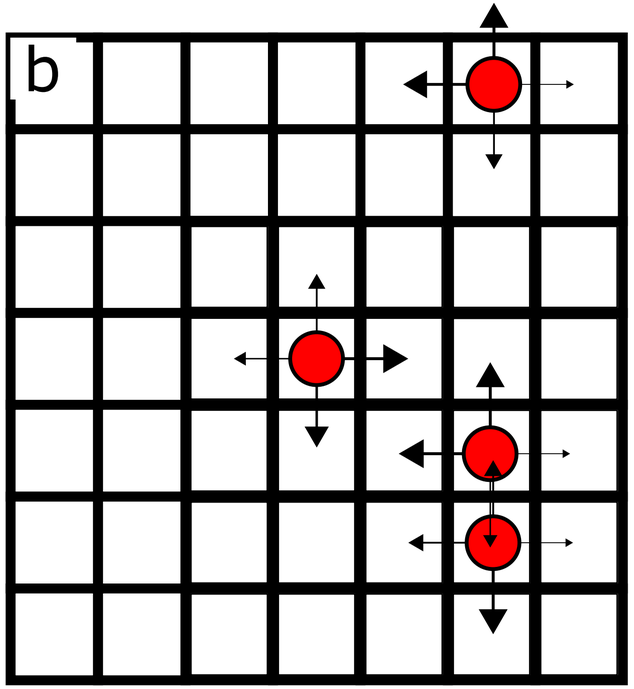
Examples:

Large arrow=Large Q-value for that action

4-steps to elimination



Shortest total path (MWPM)



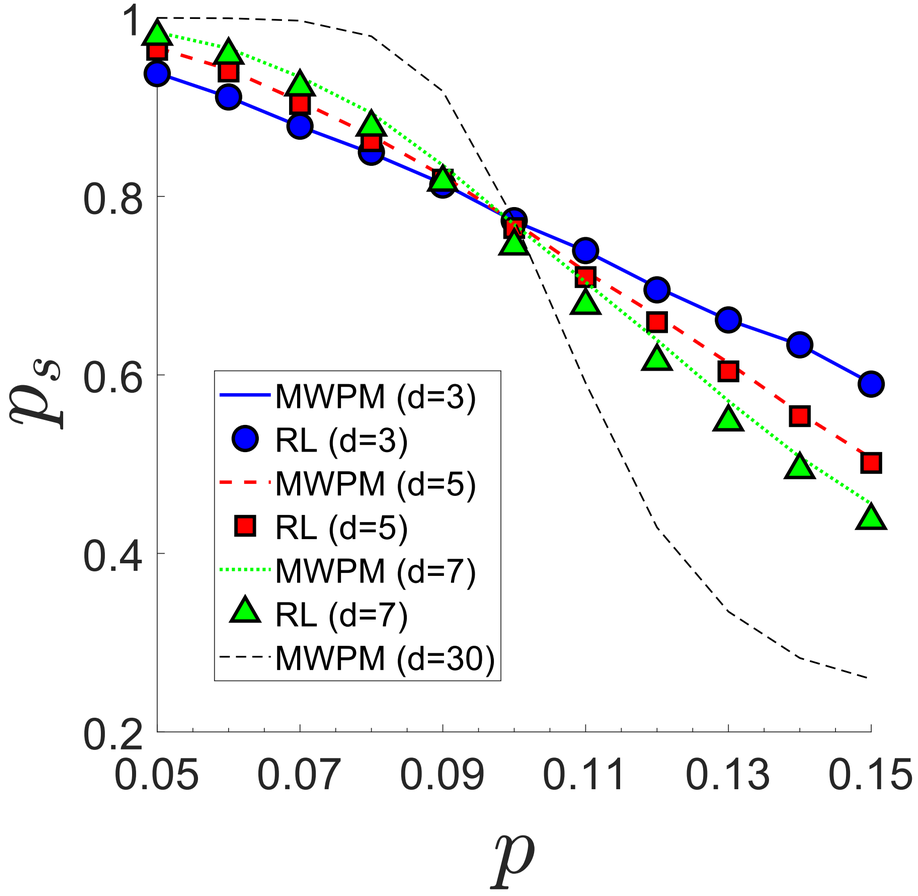
$$R = -1 - \gamma - \gamma^2 - \gamma^3 = -3.62$$

$$\gamma = 0.95$$

(semi-) quantitatively correct Q-values

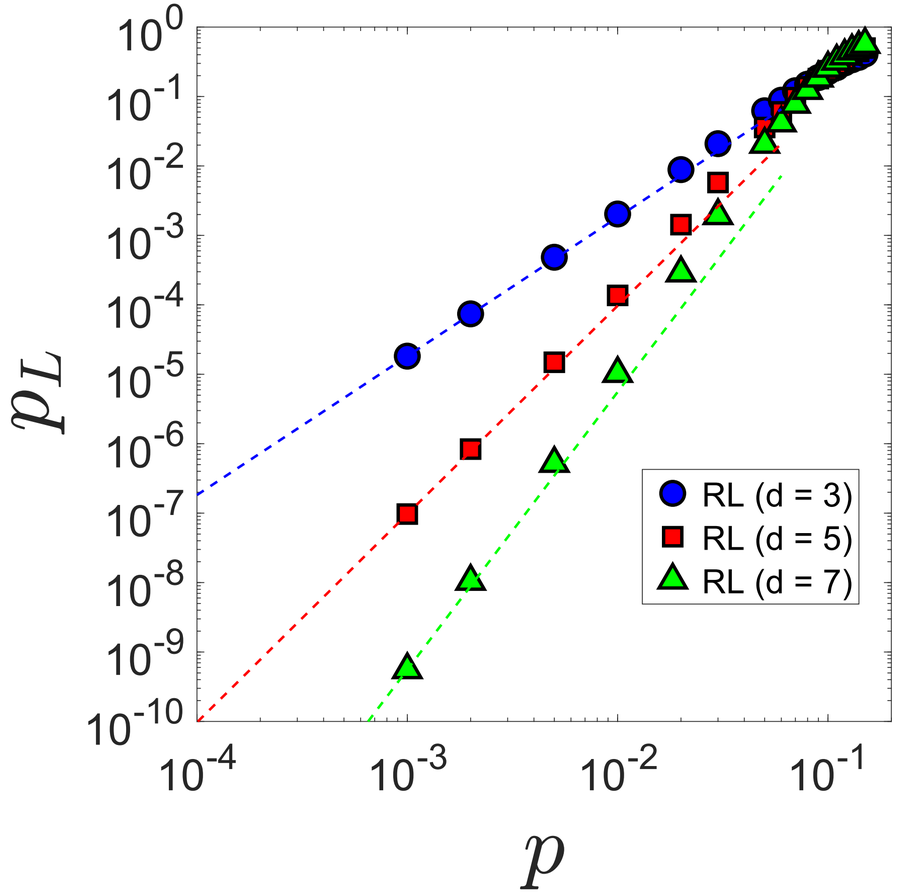
Results

Logical success-rate, large error rates close to MWPM



bit flip error rate

Logical fail-rate, small error rates identical to MWPM



Philip Andreasson, Joel Johansson, Simon Liljestrand, Mats Granath, arXiv:1811.12338

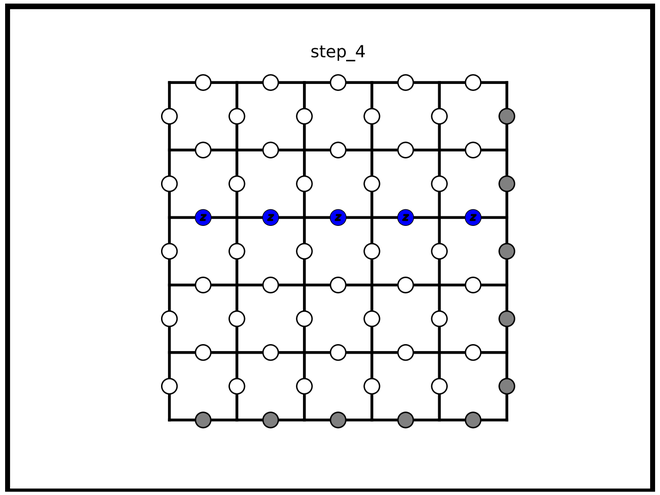
Fits asymptotic form for small p:

$$p_L = 2d\binom{d}{\lceil d/2 \rceil} p^{\lceil d/2 \rceil}$$

Depolarizing noise, work in progress

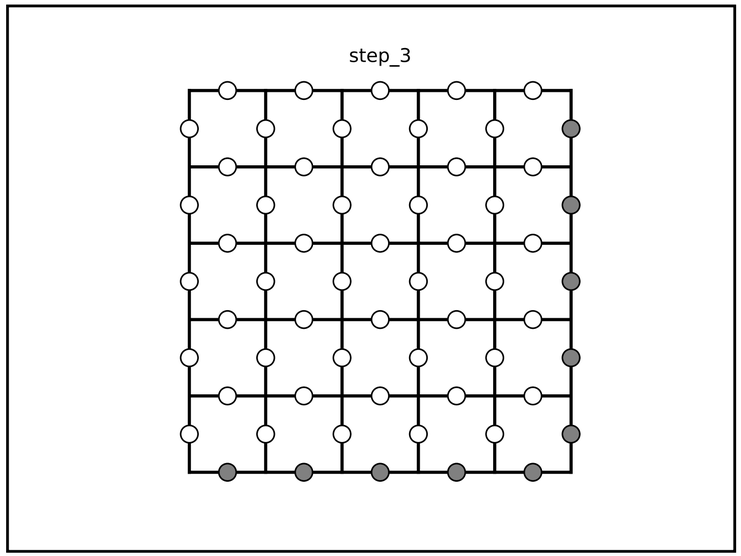
Example syndrome

MWPM



logical phase-flip

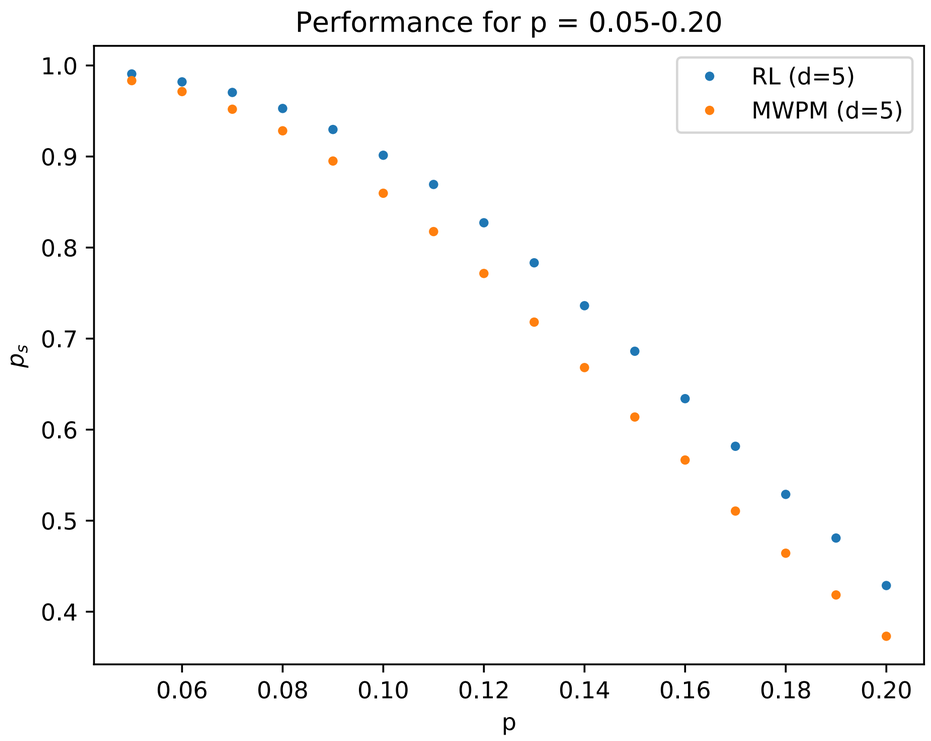
Reinforcement trained solver
reward=annihilation of complete syndrome + small intermediate reward



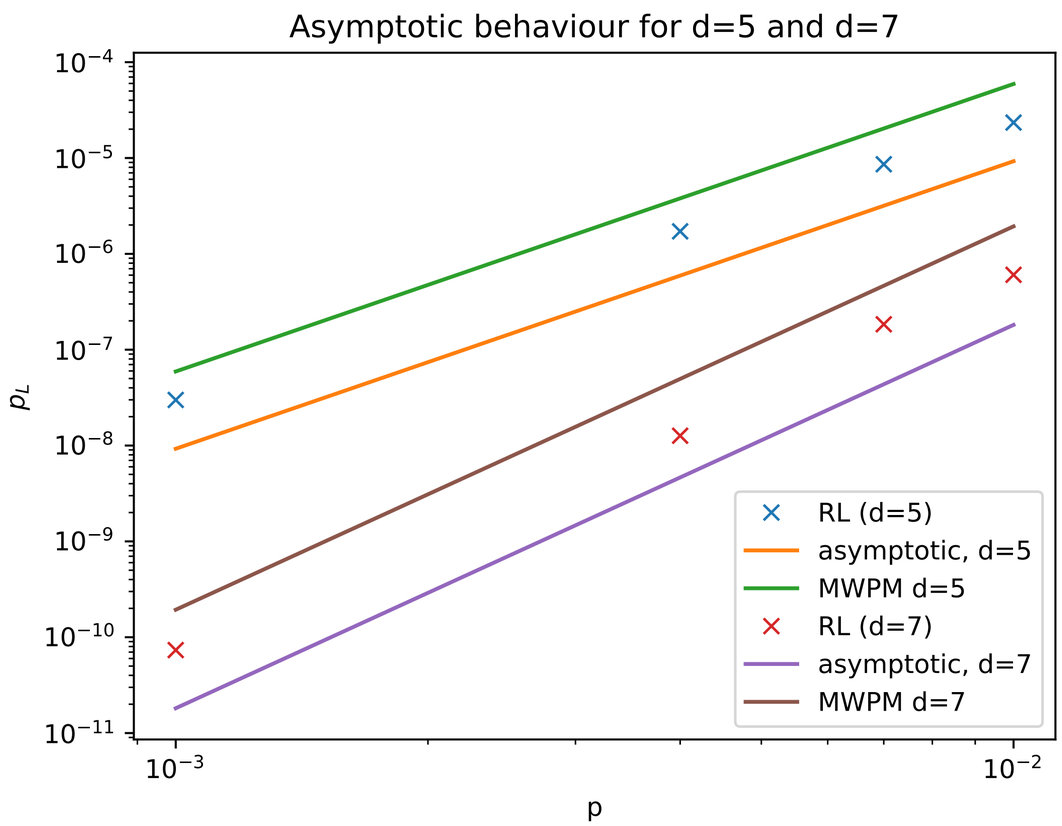
No logical operation

The agent can use Y to take advantage of correlations between bit-flip and phase-flip errors

Preliminary performance of RL solver for depolarizing noise



Outperforms MWPM



Deep Q-networks

distance 5 code

Layer (type)	Output Shape	Param #
Conv2d-1	[-1, 128, 5, 5]	2,432
Conv2d-2	[-1, 128, 5, 5]	147,584
Conv2d-3	[-1, 120, 5, 5]	138,360
Conv2d-4	[-1, 111, 5, 5]	119,991
Conv2d-5	[-1, 104, 5, 5]	104,000
Conv2d-6	[-1, 103, 5, 5]	96,511
Conv2d-7	[-1, 90, 5, 5]	83,520
Conv2d-8	[-1, 80, 5, 5]	64,880
Conv2d-9	[-1, 73, 5, 5]	52,633
Conv2d-10	[-1, 71, 5, 5]	46,718
Conv2d-11	[-1, 64, 3, 3]	40,960
Linear-12	[-1, 3]	1,731

Total params: 899,320

trained on desktop GPU for 5 hours (using PyTorch)

distance 7 code

Layer (type)	Output Shape	Param #
Conv2d-1	[-1, 200, 7, 7]	3,800
Conv2d-2	[-1, 190, 7, 7]	342,190
Conv2d-3	[-1, 189, 7, 7]	323,379
Conv2d-4	[-1, 160, 7, 7]	272,320
Conv2d-5	[-1, 150, 7, 7]	216,150
Conv2d-6	[-1, 132, 7, 7]	178,332
Conv2d-7	[-1, 128, 7, 7]	152,192
Conv2d-8	[-1, 120, 7, 7]	138,360
Conv2d-9	[-1, 111, 7, 7]	119,991
Conv2d-10	[-1, 104, 7, 7]	104,000
Conv2d-11	[-1, 103, 7, 7]	96,511
Conv2d-12	[-1, 90, 7, 7]	83,520
Conv2d-13	[-1, 80, 7, 7]	64,880
Conv2d-14	[-1, 73, 7, 7]	52,633
Conv2d-15	[-1, 71, 7, 7]	46,718
Conv2d-16	[-1, 64, 5, 5]	40,960
Linear-17	[-1, 3]	4,803

Total params: 2,240,739

trained on desktop GPU for 12 hours

Unnecessarily deep?

Conclusions

Deep Q-learning works well for error correction on *toric* code.
Can match or even outperform MWPM (for moderate code distance)

But, does require quite deep Q-networks

Periodic boundaries important for our approach.

Future challenges:

- **Larger code distances**
- **Improve reward scheme, use actual success or failure of error correction**
- **Include syndrome measurement error.** (R. Sweke et al, arXiv:1810.07207)
- **Surface code with boundaries. (Tougher due to lack of translational invariance)**

Philip Andreasson, Joel Johansson, Simon Liljestrand, Mats Granath, arXiv:1811.12338

Mattias Eliasson, David Fitzek, MG, in progress