Lecture 3: ML & string landscape

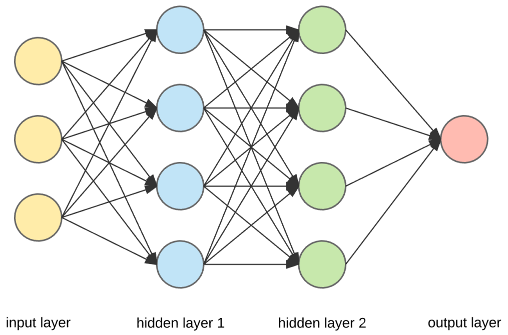
Reinforcement Learning

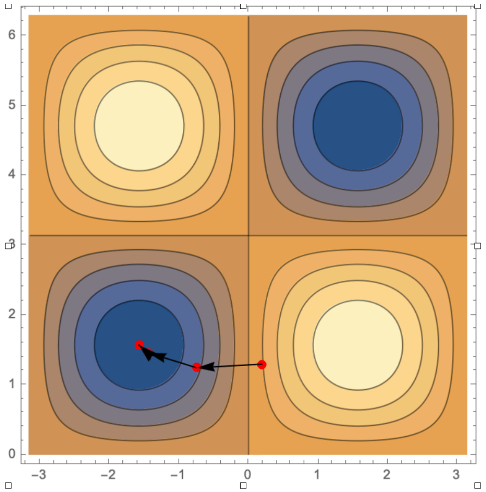
Magdalena Larfors, Uppsala University
Nordita Winter School 2024

Summary of lecture 1-2

Lecture 1

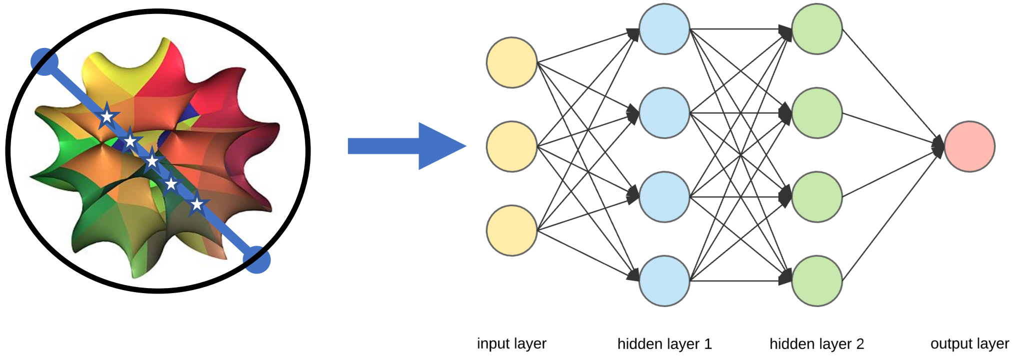
- Universal function approximator
- Supervised learning (labelled data)
- SGD, Backpropagation





Lecture 2

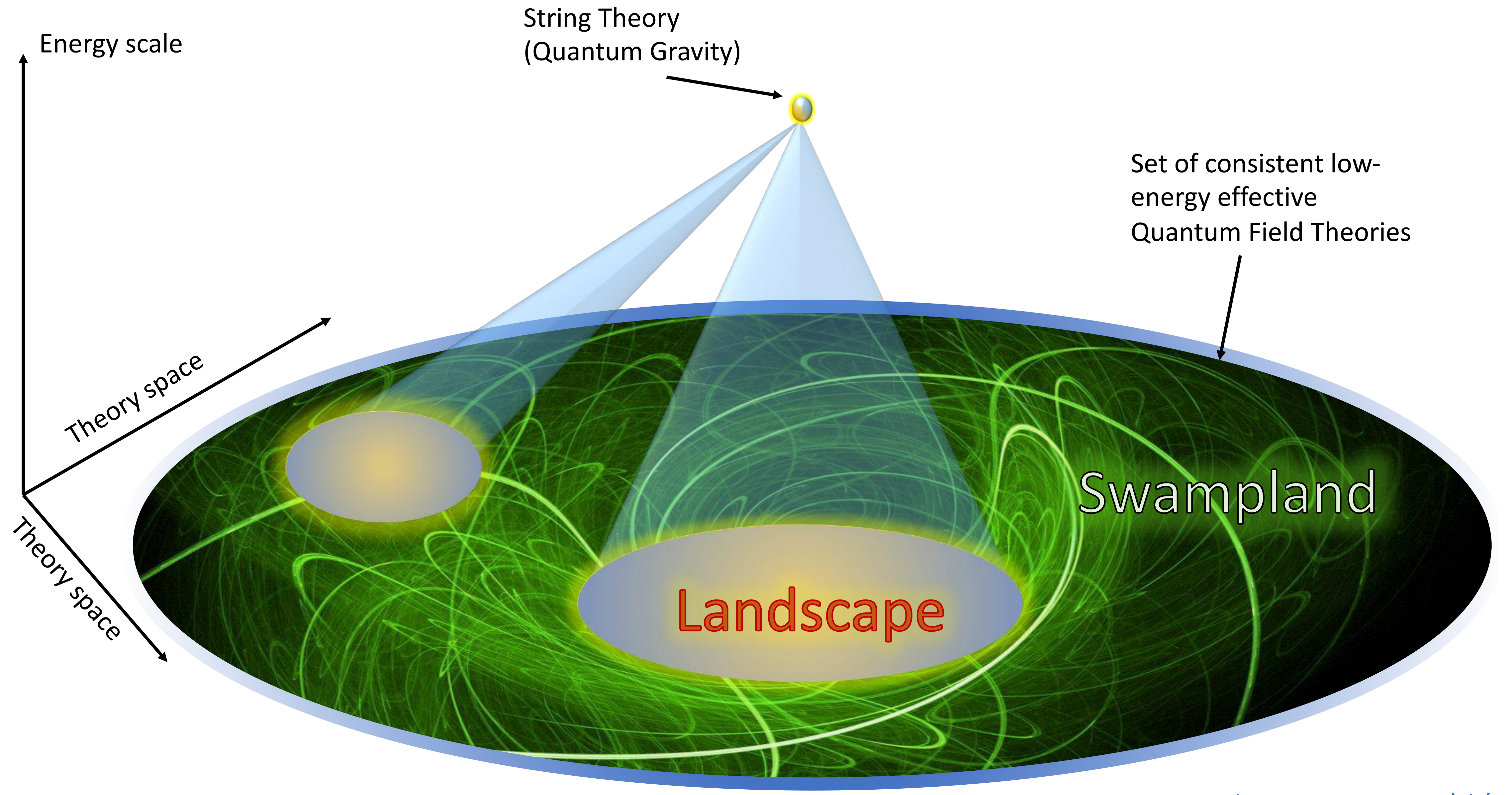
- Simple NNs learn tricky geometry
- Semi-supervised learning (data+constraints)
- Custom loss functions



Outline: ML methods for Landscape searches

- Motivating problems
  - Finding our Universe in the string theory Landscape
- Reinforcement Learning
  - Simple examples
  - States and actions
  - Rewards and returns
  - Goals and policies
- RL finds Standard-like Model (SLM) physics from heterotic line bundles

- Summary, reading and tutorials



Picture source: Palti (19)

String compactifications

- Pick your favourite version of string theory:
  M-theory, F-theory, type IIA/IIB, heterotic, …
- Pick a SUSY-preserving compact geometry to get 4d:
  G2 mfd, CY 4-fold, CY 3-fold, toroidal orbifold, SU(3) structure mfd, …
- Pick extra ingredients so you get the Standard Model (or GUT, …)
  singularities, singular fibers, branes/O-planes, vector bundles, …

- Then compute couplings, mass terms, stabilize moduli, break SUSY, …

Many choices to make!

Heterotic compactifications on CYs

- Pick your favourite version of string theory: heterotic $E_8 \times E_8$
- Pick a SUSY-preserving compact geometry to get 4d: CY 3-fold X
- Pick extra ingredients for particle physics:
  holomorphic vector bundle $V \to X$ w. structure group in E_8
- There are still many choices!
  - 7890 CICY 3-folds — Candelas et al:88
    KS CY 3-folds from toric 4-folds — Kreuzer-Skarke:00
    473,800,776 reflexive polyhedra, each leading to 1 or more CYs
    → 10^{428} inequivalent CY 3-folds — Demirtas–McAllister–Rios-Tascon:20
  - Each CY can be paired with many different vector bundles
- A systematic scan over 10^{40} models gave 35’000 SU(5) SLMs

Anderson et al 1307.4787

String compactifications

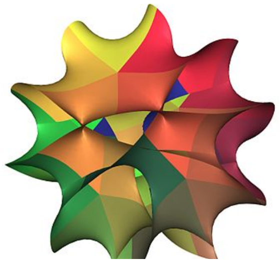
- But why should we construct 1000s of Standard-like Models?
- Surely, one good model is enough to describe our Universe?

- Yes, but "needle in a haystack" → Need good search algorithms
- Statistics of models: what is typical in string theory?
- Swampland: what cannot occur in string theory?
- Constraints are hard – working with large classes of OK models increase chance of finding a good model (whatever that means)
- Systematic scans are limited by computational resources → try ML

Can we use supervised learning?

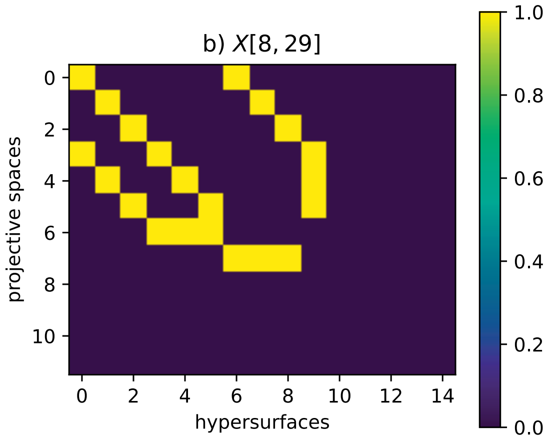
- Learn CY topology
  - Known from algebraic geometry
  - AG methods scale poorly
- Configuration matrix → Image
- Can use s-o-t-a image recognition techniques
- Current lead (inception models): >99% accuracy on $h^{1,1}$

He:17, Ruehle:17, Bull-et.al.:18,19, Klaewer–Schlechter:18, Constantin–Lukas:18, Brodie-et.al.:19, ML–Schneider:19, Erbin–Finotello:20, He–Lukas:20, Erbin-et.al.:22; ...



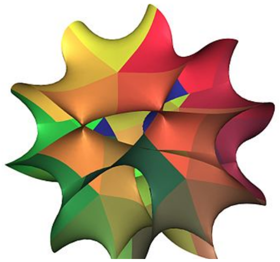
$$X \in \left[\, 4 \,\middle\|\, 5 \,\right]$$

$$X[8,29] = \left[\begin{array}{c||cccccccccc} 1 & 1 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 1 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 2 & 1 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 1 \\ 2 & 0 & 1 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 1 \\ 2 & 0 & 0 & 1 & 0 & 0 & 1 & 0 & 0 & 0 & 1 \\ 2 & 0 & 0 & 0 & 1 & 1 & 1 & 0 & 0 & 0 & 0 \\ 2 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 0 \end{array}\right]^{8,29}_{-42}$$



Can we use supervised learning?

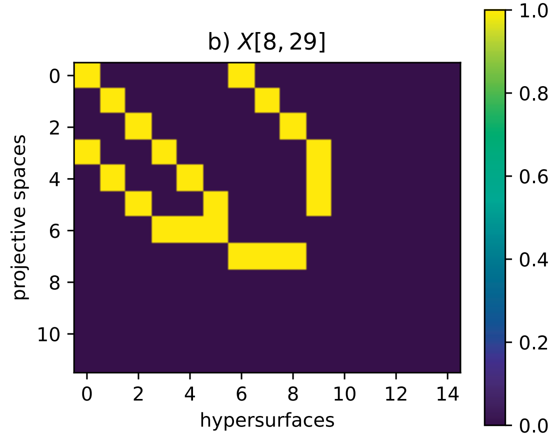
- Current lead (inception models):
  >99% accuracy on $h^{1,1}$



$$X \in [\ 4 \,\|\, 5\ ]$$

- So this would give (jumping steps) 99% of the Standard Model?

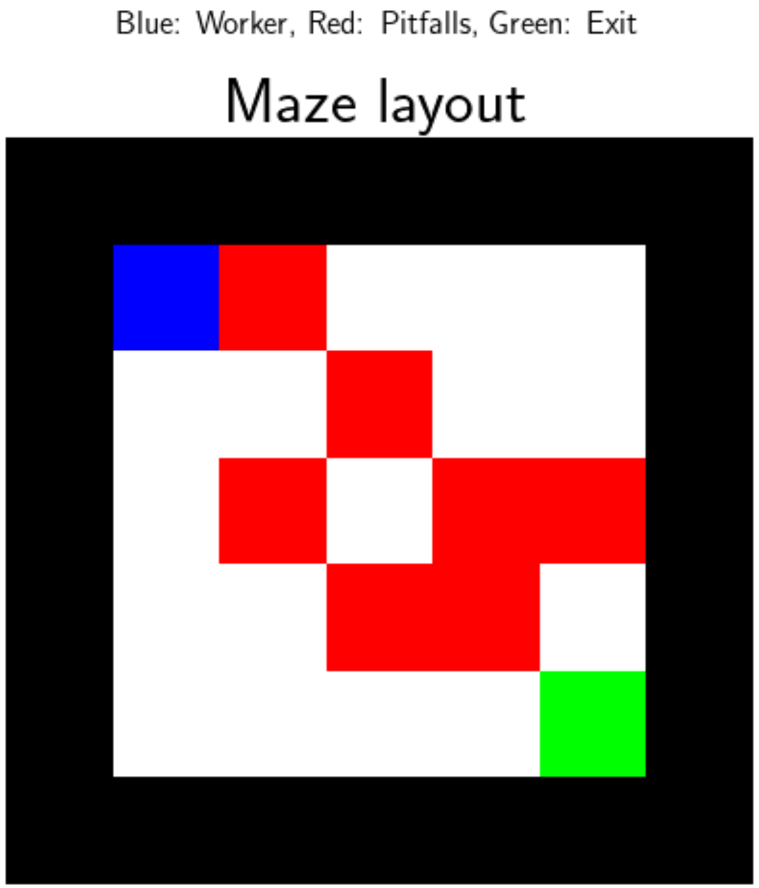
$$X[8,29] = \left[\begin{array}{c||cccccccccc} 1 & 1 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 1 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 2 & 1 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 1 \\ 2 & 0 & 1 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 1 \\ 2 & 0 & 0 & 1 & 0 & 0 & 1 & 0 & 0 & 0 & 1 \\ 2 & 0 & 0 & 0 & 1 & 1 & 1 & 0 & 0 & 0 & 0 \\ 2 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 0 \end{array}\right]^{8,29}_{-42}$$

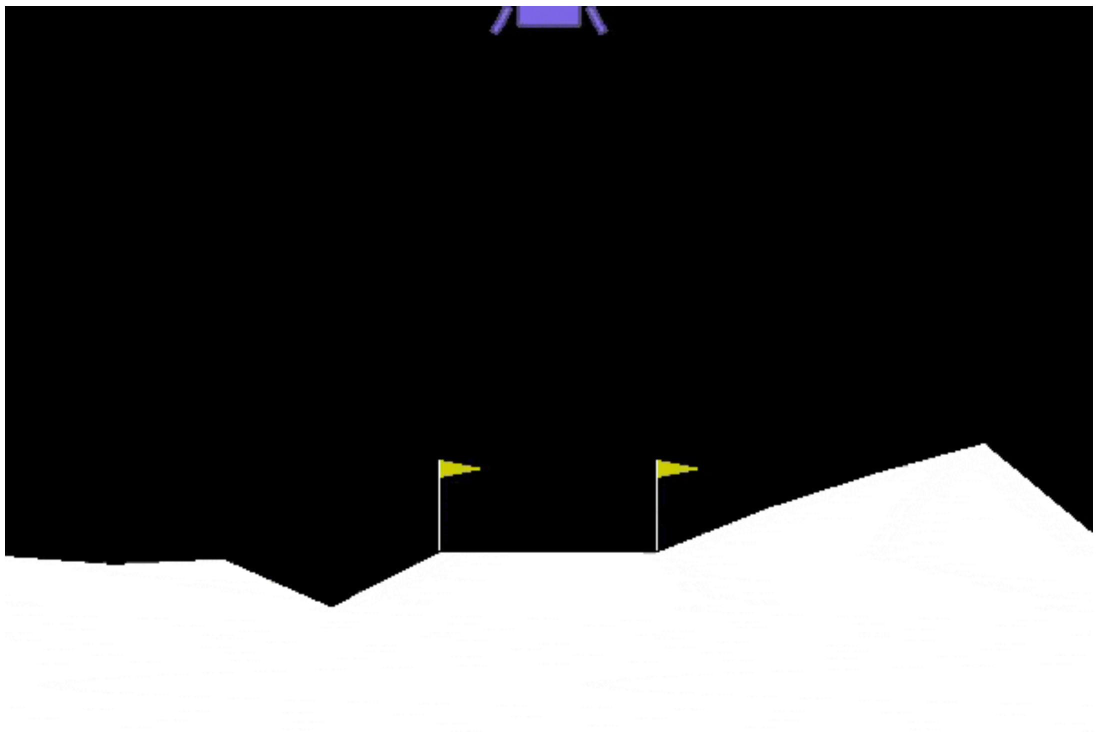


- What does that mean?

Intro to Reinforcement Learning

Reinforcement Learning ♡ Games



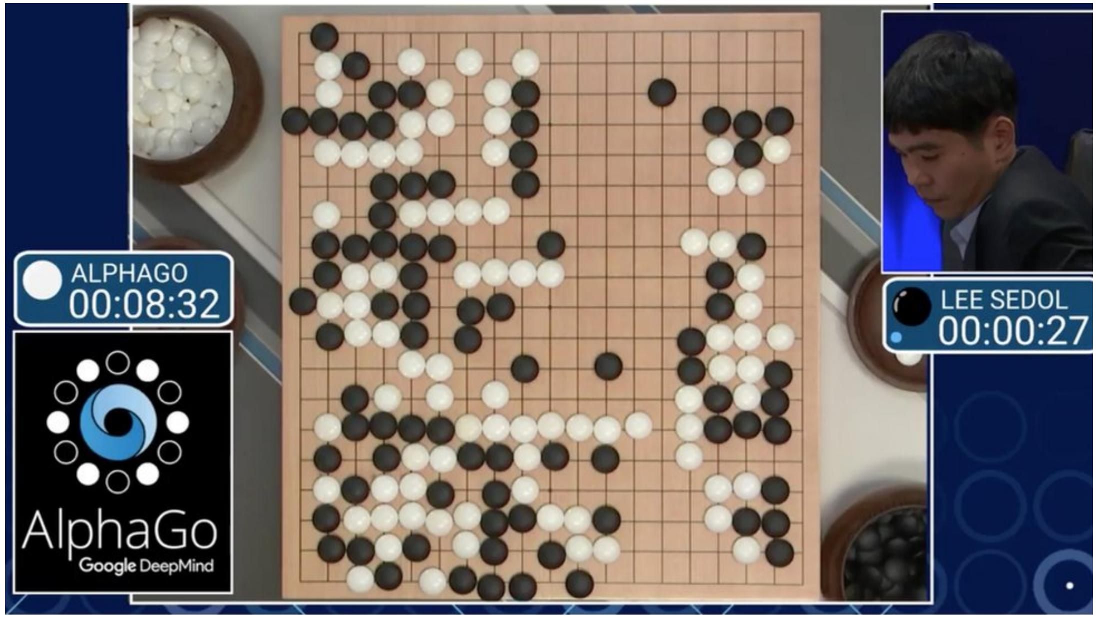


Reinforcement Learning

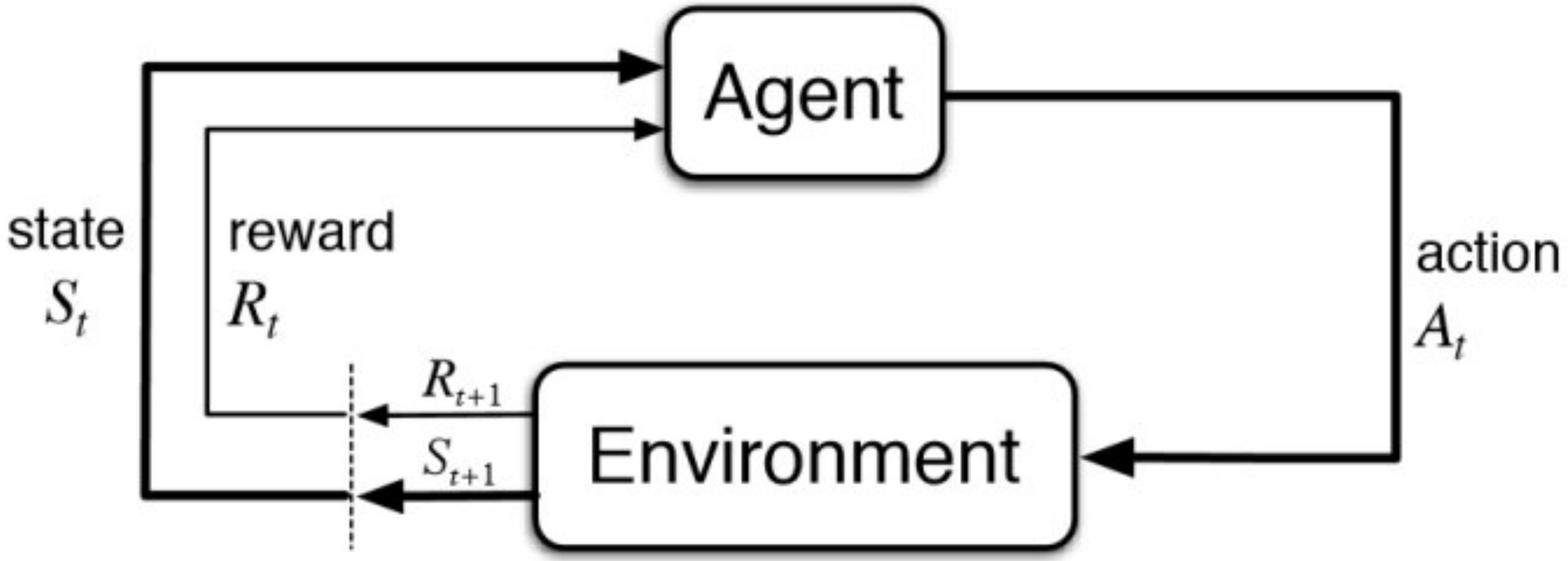
- Idea: agent learns to win a game (solve an environment) by receiving rewards (pos & neg)

- Solves large environments AlphaGo Silver et al (Science, 2018)
- Learns from imperfect info OpenAI wins DotA 2

  Berner et al 1912.06680

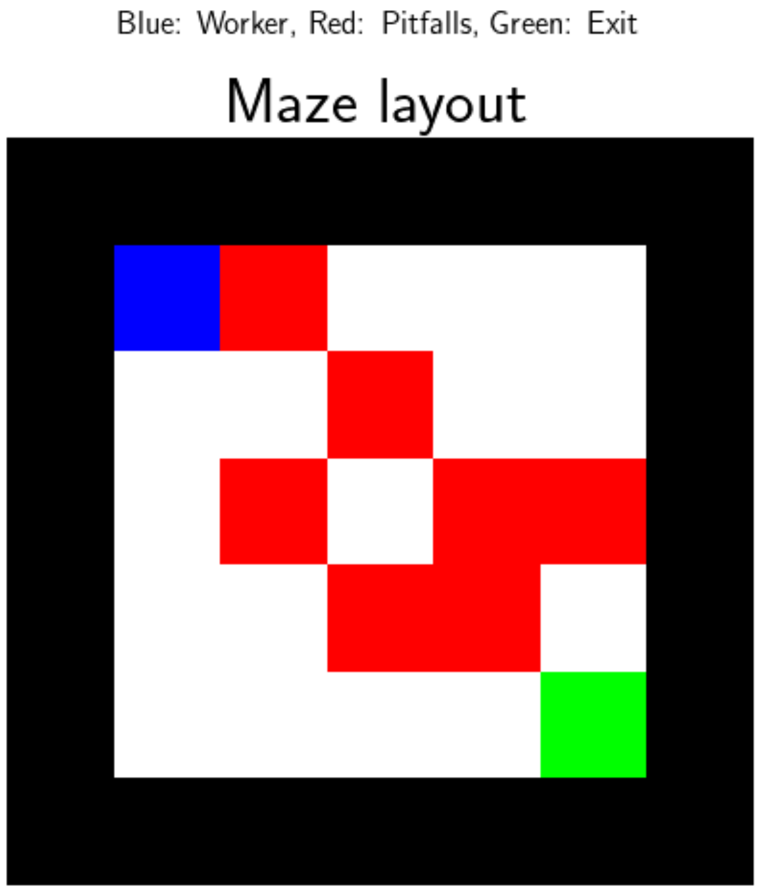


Reinforcement Learning



RL: solving a maze

- Environment = states & actions
- States: positions {(1,1),(1,2),…}
- Action: 1 step {W,S,E,N}
- Rewards:
  - e.g. -1 for each step, -5 for hitting wall, -10 for pitfall, +100 for exit, …
- The agent must develop a policy for which action to take, given the info available in the current state
- Explore vs exploit
- Try it out in tutorial!



RL: terminology

- Environment, $\mathcal{E}$, is set of states $\mathcal{S}$ and set of actions $\mathcal{A}$
- States $\mathcal{S} = \{s_i\}$: possible configurations (continuous/discrete, finite/infinite)
- Actions $\mathcal{A} = \{a_i\colon \mathcal{S} \to \mathcal{S}\}$: transitions between states
- Terminal states : no action possible; search ends here (pitfall or exit of maze)
- Episode: Sequence of states and actions that ends in a terminal state

$$E = \{(s_1, a_1), (s_2, a_2), \dots, (s_n, \emptyset)\}$$

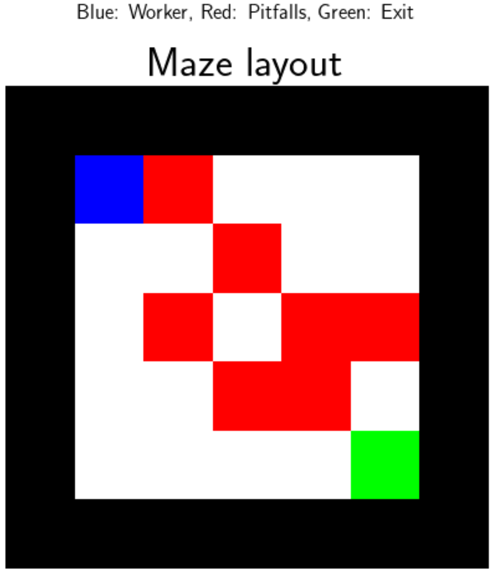
RL: terminology

- Policy, $\pi: \mathcal{S} \rightarrow \mathcal{A}$: given current state, determines action
  - Deterministic or probabilistic
  - Determined by expected value of action
- Reward: feedback to agent, depends on state and action
- Return (discounted accumulated future reward, depends on policy)

$$G_t = R_{t+1} + \gamma R_{t+2} + \ldots = \sum \gamma^k R_{t+k+1}$$

- Goal: find policy that maximizes return

RL with/without ML: back to the maze



- Markov decision problem
- Find policy given info of the value of states and actions
- For a small maze, can solve algorithmically
  - Try 1: {(1,1;E)} gives $G = -10$
  - Try 2: {(1,1;S),(2,1;E),(2,2;E)} gives $G = -1 - \gamma - 10\gamma^2$
  - …
- Collect info of {states, actions, returns} in order to determine policy
- Requires memory
- Unfeasible for large mazes, or more complicated problems

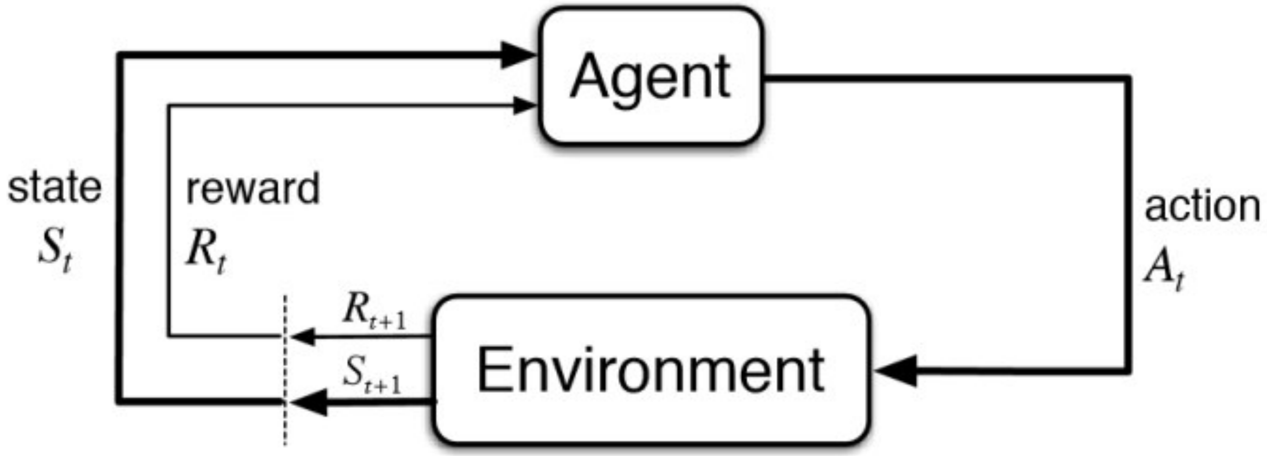
RL: enter the neural nets

- State value function: $V_\pi(s) = E(G_t \mid s = s_t)$
  Expected return from current state onward when following policy π
- Action value function: $q(a, s) = E(G_t \mid s = s_t, a = a_t)$
  Expected return from picking an action in a given state
- Policy, state value and action value functions are interdependent.
- We seek estimators for best state value function, action value function, and policy; use neural nets
  - Policy: NN observes state, predicts action
  - State value function: NN observes state, predicts expected return
  - Action value function: NN oserves state and action, predicts expected return

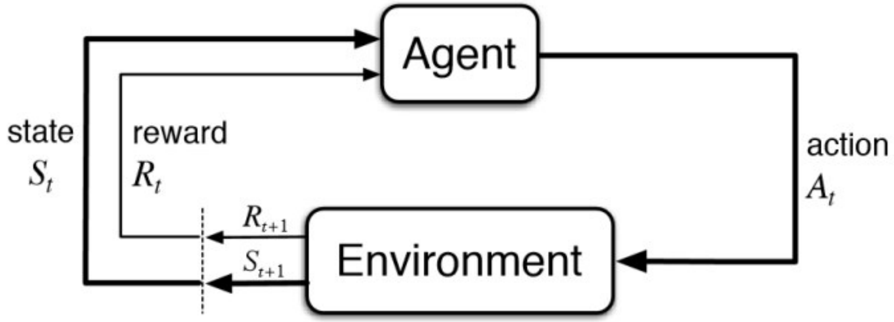
RL: summary

- Agent observes state in environment and take action following policy
- Feedback: reward and new state
- Agent updates policy and state/action value functions
- Agent observes new state… etc

- NNs can estimate policy and value functions



RL: algorithms & libraries



- NNs can estimate policy and state/action value functions
- REINFORCE:
  - NN for policy $\pi(a_t|s_t;\theta_\pi)$
  - parameter update by gradient ascent w.r.t (optimal) state value $v_\pi^*(s)$
  - this requires full reward information of the whole episode
- Actor-Critic:
  - NN as **Actor** to update policy $\pi(a_t|s_t;\theta_\pi)$
  - NN as **Critic** to update state value function $V(s_t;\theta_v) = E[G_t|s = s_t]$
  - Update parameters θ_π, θ_V using gradient ascent from

    Advantage function $A = R_i - V(s_i;\theta_V)$

    Cross-entropy over actions

RL for heterotic model building

3 building blocks for heterotic SLMs

- Calabi Yau manifold $\mathcal{M}$.
- Discrete symmetry Γ
  (for Wilson line GUT breaking).
- Line bundle sum $V = \bigoplus L_a$.
- Explored systematically → 35 000 SLMs
  Anderson et al (1106.4804,1202.1757,1307.4787)

- Example

$$\mathcal{M}_{5302} = \left[\begin{array}{c||ccc} 1 & 0 & 1 & 1 \\ 1 & 0 & 1 & 1 \\ 1 & 1 & 1 & 0 \\ 1 & 1 & 1 & 0 \\ 1 & 1 & 0 & 1 \\ 1 & 1 & 0 & 1 \end{array}\right]^{6,30}_{-48} \qquad |\Gamma| = 2$$

$$V = \begin{bmatrix} -1 & 0 & 0 & 0 & 1 \\ 4 & -3 & -1 & 0 & 0 \\ 0 & 0 & -1 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & -1 \\ 1 & 1 & 0 & -2 & 0 \end{bmatrix}$$

NB: configuration encoded in simple integer matrices!

RL heterotic SLMs

ML-Schneider:2003.04817
R. Schneider PhD thesis, 2022

Heterotic string compactification with three ingredients

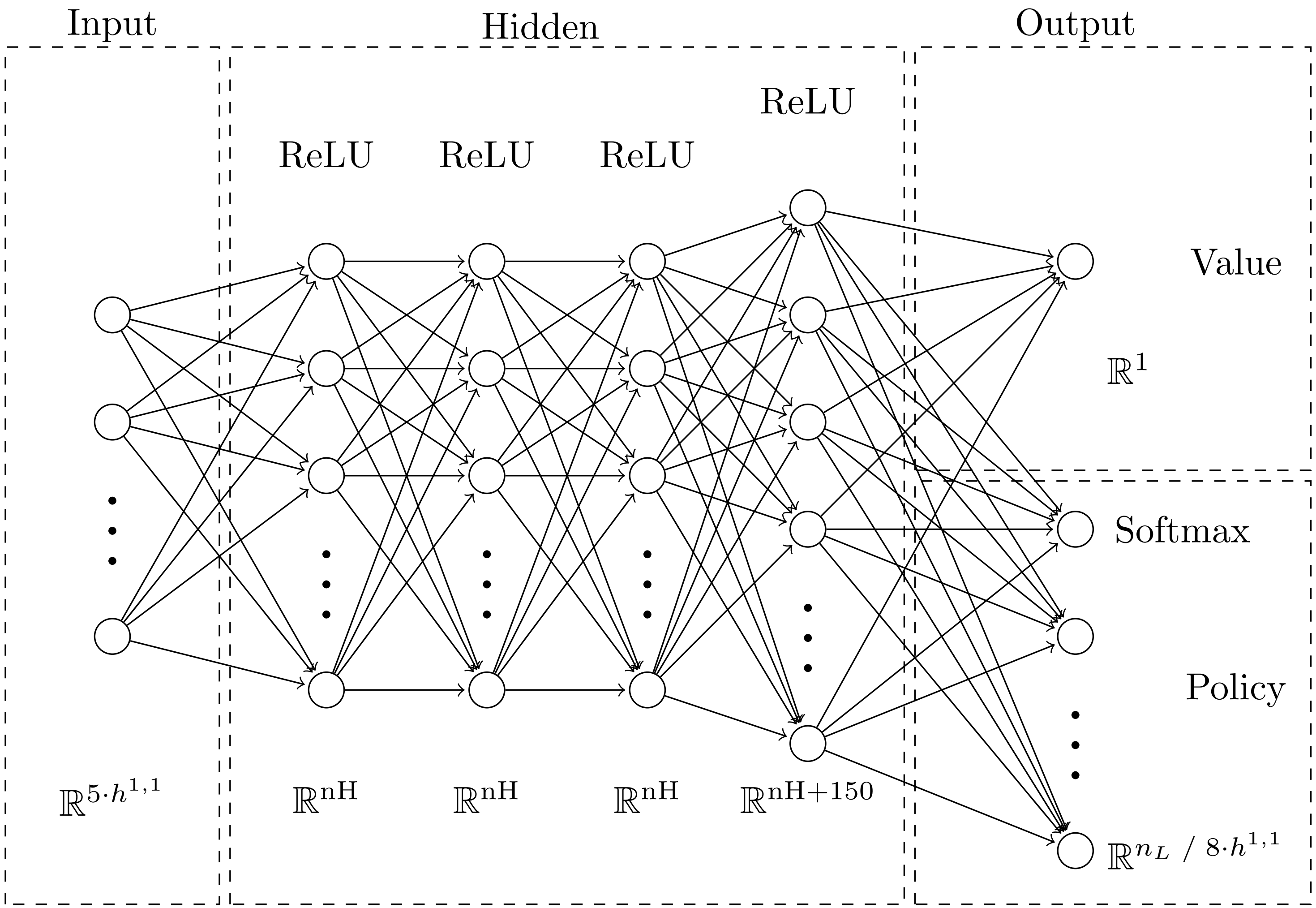
- Calabi Yau manifold $\mathcal{M}$.
- Freely acting discrete symmetry Γ.

Keep fixed

- Line bundle sum $V = \bigoplus L_a$. → Environment for RL exploration

- The environment $\{V = \bigoplus L_a\}$ are just integer matrices
  In our paper, we set up two environments, with actions
  - Stacking: precompiled list of slope stable L_a stacked and then replaced
  - Flipping: initiate randomly and then flip individual entries
  - Inspired by RL of intersecting brane models Halverson–Nelson–Ruehle:1903.11616

RL: implementation and libraries



Use A3C:

Asynchronous Advantage Actor Critic

Mnih et al (1602.01783)

ChainerRL

Open AI gym

RL heterotic SLMs

To qualify as a heterotic SLM, the bundle must satisfy physical constraints.

$\rightarrow$ Translate constraints to reward structure.

RL heterotic SLMs: reward structure

condition	reward
vanishing first Chern class	trivial
vanishing line bundle slope (3.28)	2
index constraint, three fermion generations (6.8)	10^2
Bianchi identity (6.4)	10^5
no Higgs triplets (6.12)	10^5
existence of Higgs doublets (6.12)*	10^6
no antigeneration (6.11)*	10^7
full stability (6.6)	10^7

- Topological constraints.
- Some immediate; others need AG;
- Automated: pyCICY, cohomcalg

Environments: Stacking

Precompile list L of n_{line} slope stable line bundles with $-3|\Gamma| \leq \text{index}(L_a) \leq 0$

Stack four of these, and adjust L_5 to satisfy $c_1(V) = 0$

$\longrightarrow$ *Constraints 1, 2, 3 are automatic*

States: The line bundle sum V. Hence $S_t \in \mathbb{Z}^{(5,\text{nProj})}$.

Actions: Pick $L_a \in L$ and replace one of L_{1-4}.

# of configurations: $n_{\text{conf}} = n_{\text{line}}^4$.

Example: $(\mathcal{M}_{5302},\ q_{\text{max}} = 2,\ |\Gamma| = 2)$ gives $n_{\text{line}} = 2890$ and $n_{\text{conf}} \approx 7 \cdot 10^{13}$.

$$\begin{bmatrix} -1 & 0 & 0 & 0 & 1 \\ 2 & 0 & -1 & -1 & 0 \\ -2 & -1 & 0 & -2 & 5 \\ 0 & -1 & 0 & 2 & -1 \\ 0 & 2 & -2 & 2 & -2 \\ 2 & 2 & 1 & 0 & -5 \end{bmatrix} \xrightarrow{A_t} \begin{bmatrix} -1 & -2 & 0 & 0 & 3 \\ 2 & -2 & -1 & -1 & 2 \\ -2 & 0 & 0 & -2 & 4 \\ 0 & 2 & 0 & 2 & -4 \\ 0 & 0 & -2 & 2 & 0 \\ 2 & 2 & 1 & 0 & -5 \end{bmatrix}$$

Environments: Flipping

No precompiled list, instead "flip" individual charges in L_{1-4}

Still adjust L_5 to satisfy $c_1(V) = 0$

$\longrightarrow$ *Constraint 1 is automatic*

States: The line bundle sum V. Hence $S_t \in \mathbb{Z}^{(5,\mathrm{nProj})}$.

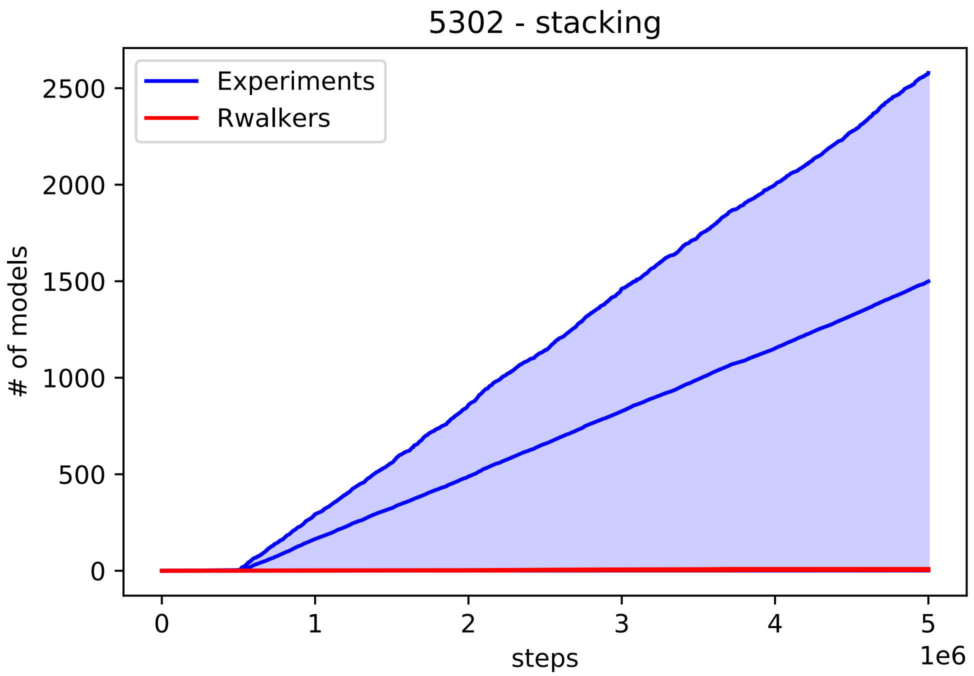
Actions: Pick a charge q_i^j and add ± 1. Thus $A_t \in \{1, ..., 4 \cdot 2 \cdot \mathrm{nProj}\}$.

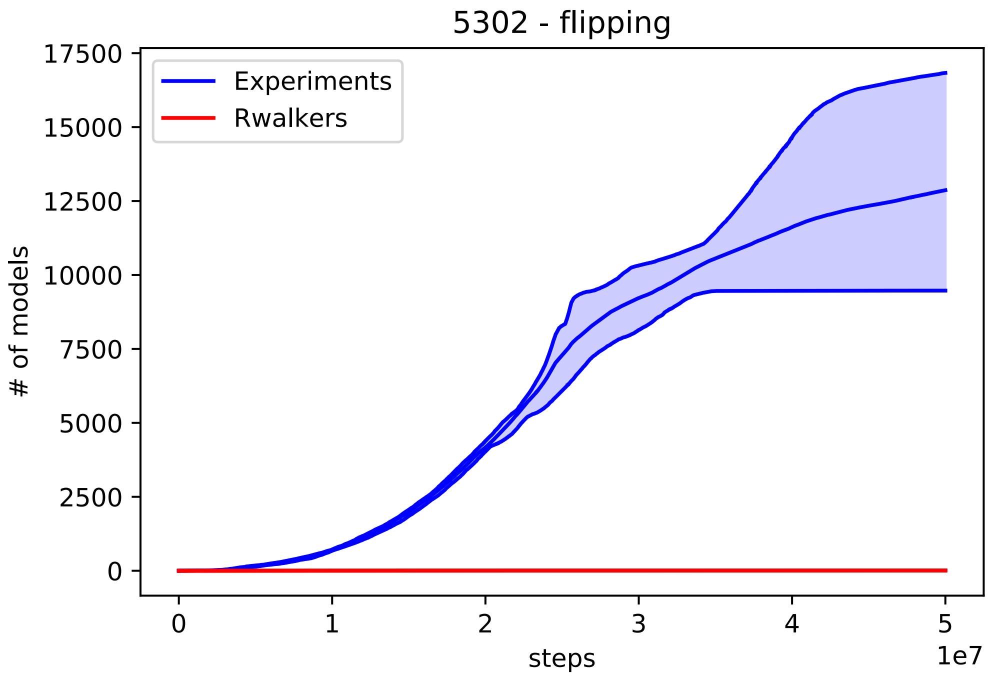
# of configurations: $n_{\mathrm{conf}} = (2 \cdot q_{\mathrm{max}} + 1)^{4 \cdot h^{1,1}}$.

Example: $(\mathcal{M}_{5302},\ q_{\mathrm{max}} = 2,\ |\Gamma| = 2)$ gives $n_{\mathrm{conf}} \approx 5 \cdot 10^{16}$.

$$\begin{bmatrix} 1 & 1 & -1 & 0 & -1 \\ -1 & 0 & 1 & 0 & 0 \\ 1 & 1 & 0 & -1 & -1 \\ 1 & 1 & -1 & -1 & 0 \\ -1 & 1 & 0 & 0 & 0 \\ -1 & 1 & 0 & 0 & 0 \end{bmatrix} \xrightarrow{A_t} \begin{bmatrix} 2 & 1 & -1 & 0 & -2 \\ -1 & 0 & 1 & 0 & 0 \\ 1 & 1 & 0 & -1 & -1 \\ 1 & 1 & -1 & -1 & 0 \\ -1 & 1 & 0 & 0 & 0 \\ -1 & 1 & 0 & 0 & 0 \end{bmatrix}$$

Stack or Flip?





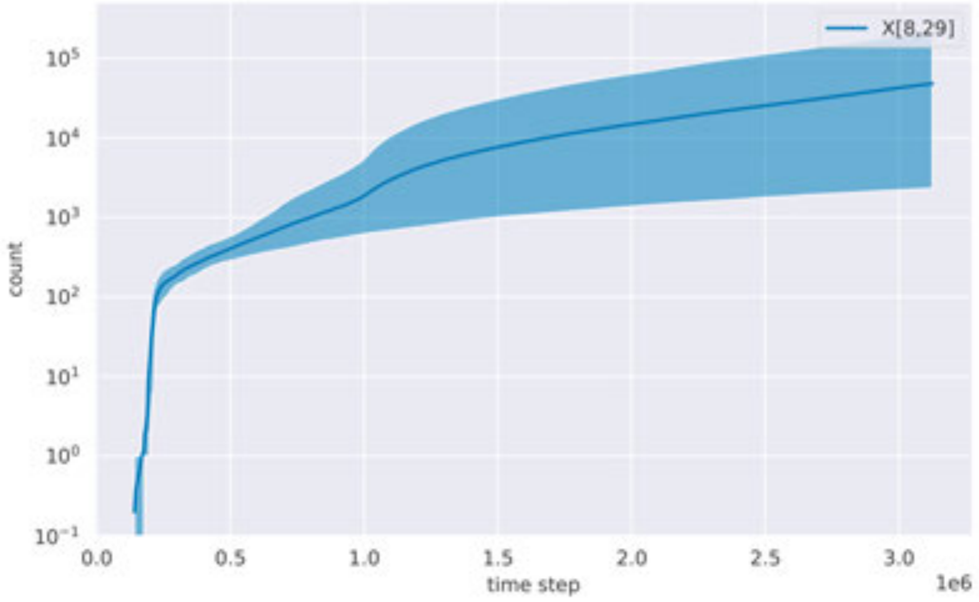
Stack or Flip?

- Stacking close to human derived strategy of systematic scan
  - Runtime about 50 minutes on 32 cores
  - Moderately outperform random walker (factor 3-20)
  - Gets stuck in local minima –→ low number of unique models.
- Flipping strategy different from systematic scan
  - Runtime about 3.5 hours on 32 cores
  - Rapid increase in performance followed by flattening at late times
  - Significantly outperform random walker (factor 300-1700)
  - Large number of unique models.

RL goes beyond $h^{1,1} = 7$ (unprobed by systematic scans)

$$X[8,29] = \left[\begin{array}{c||cccccccccc} 1 & 1 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 1 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 2 & 1 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 1 \\ 2 & 0 & 1 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 1 \\ 2 & 0 & 0 & 1 & 0 & 0 & 1 & 0 & 0 & 0 & 1 \\ 2 & 0 & 0 & 0 & 1 & 1 & 1 & 0 & 0 & 0 & 0 \\ 2 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 0 \end{array}\right]^{8,29}_{-42}$$

R. Schneider PhD thesis, 2022

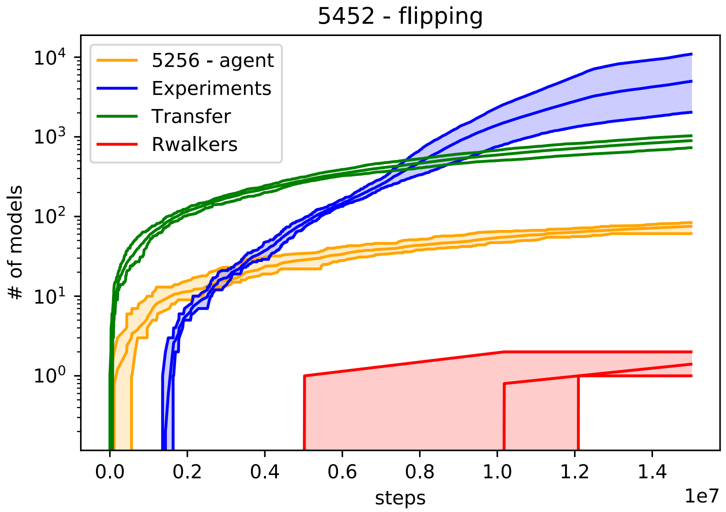


run	all	unique
1	3576	2863
2	9577	4451
3	2296	1973
4	3554	2759
5	220071	9289
total	-	14374

RL design & transfer learning

- RL is slow to start
- Pretraining agent possible:
  - basic scan, with fewer constraints/rewards
  - smaller set of bundles
- Benefit: pre-trained network could then be applied in full setting

- Transfer learning
- In 2003.04817 used pretrained agent from $\mathcal{M}_{5265}$ on $\mathcal{M}_{5452}$



Summary

Reinforcement learning:

- Idea: agent learns to win a game (solve an environment) by receiving rewards (pos & neg)
- Versatile method, can cope with large environments (discrete/cont.) and incomplete info
- Govern by Policy, $\pi: \mathcal{S} \to \mathcal{A}$: given current state, determines action
    - Deterministic/probabilistic; Determined by expected value of action
- State and action value functions: expected return from state/action
- NNs can be used to estimate policy and value functions –> ML libraries ChainerRL, OpenAI gym
- Packages for string/math-related RL
  https://github.com/robin-schneider/gymCICY
  https://github.com/ruehlef/ribbon

Plan for afternoon studies

- Reading:
  - F. Ruehle. “Data science applications to string theory”, ch 8
    https://www.sciencedirect.com/science/article/pii/S0370157319303072
  - R. Schneider “Heterotic Compactifications in the Era of Data Science”, ch. 6,7
    http://uu.diva-portal.org/smash/record.jsf?pid=diva2%3A1649343&dswid=-2157
  - Heterotic model building with RL:
    M. Larfors & R. Schneider arXiv:2003.04817 S. Abel et al 2110.14029 2306.03147
- Online tutorials:
  - Intro to RL by Callum Brodie
    https://colab.research.google.com/github/callum-ryan-brodie/oxford-ml-physmath-school/blob/main/oxford_ml_physmath_school_notebook_2.ipynb#scrollTo=qDF-j8opXEaH
  - RL of heterotic LB models by Robin Schneider
    https://github.com/robin-schneider/gymCICY/blob/master/agents/Tutorial.ipynb