RL and optimization

- All said falls into the domain of optimization:
  - ➡ An optimiser tries to find the arguments of a function to maximise the function value (optimization is greedy!)
  - ➡ RL algorithms look to find a mapping (the policy) from states to actions maximising the expected cumulative reward rather than just a single optimal function value
    - If parametric function approximation is used, we try to find the values of the parameters of the approximated function (either a value function or the policy directly) to obtain this mapping (this a classical optimisation problem).
- RL is comparable to calculus of variation (its origin is in classical mechanics - HJB equation) instead of function optimization

Optimization

maximise$_{\{A_i\}} \sum_{t=0}^{T} R(S_t, A_t, W_t)$

subject to: $S_{t+1} = f(S_t, A_t, W_t)$

RL

maximise$_{\pi_t} \mathbb{E}_{W_t}[\sum_{t=0}^{T} R_t(S_t, A_t, W_t)]$

subject to: $S_{t+1} = f(S_t, A_t, W_t)$

$\boxed{A_t = \pi_t(S_t, S_{t-1}, \ldots)}$

Feedback structure takes noise into account

Often optimization is performed only for one step horizon:

maximise$_a R(\,\cdot\,, a, W_t)$

Possible solutions

- Use a high fidelity model
  - ➡ Usually unavailable…
- Development of faster algorithms
  - ➡ SAC (very efficient maximum entropy algorithm)
  - ➡ TD3 (simple and efficient tricks accelerate DDPG-Style algorithms)
- Use data efficient method as MBRL
  - ➡ Hard to handle, low computational efficiency
- Reuse von prior knowledge to accelerate RL
  - ➡ RL2: Fast reinforcement learning via slow reinforcement learning https://arxiv.org/pdf/1611.02779.pdf
  - ➡ Learning to reinforcement learning https://arxiv.org/pdf/1611.05763.pdf
  - ➡ Model-agnostic meta-learning http://proceedings.mlr.press/v70/finn17a/finn17a.pdf