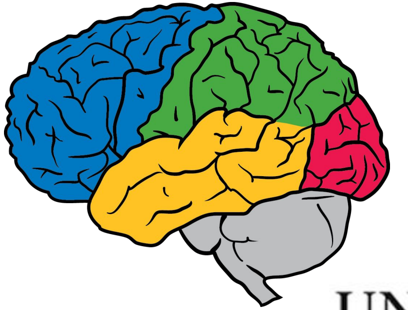
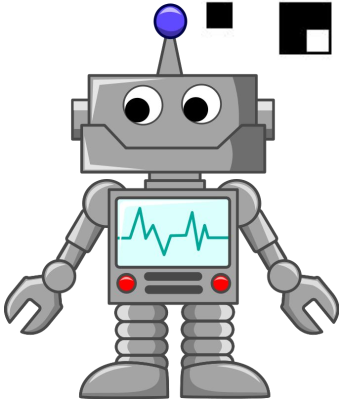
Structure is all you need - learning compressed RL policies via gradient sensing

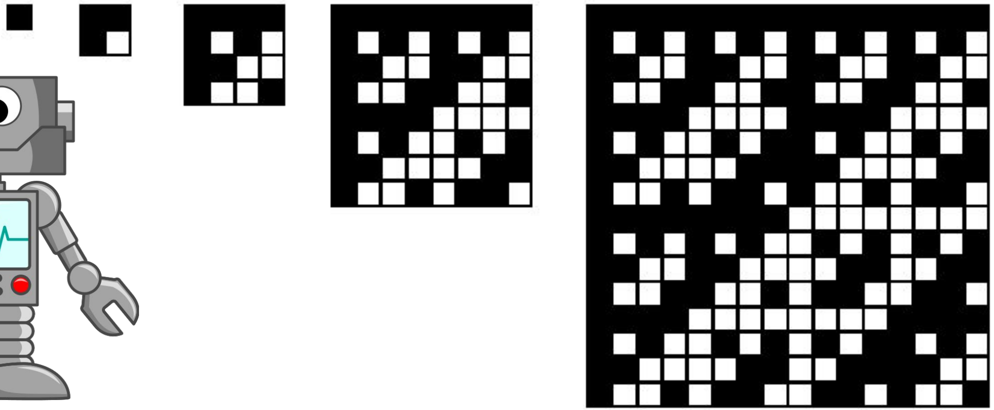


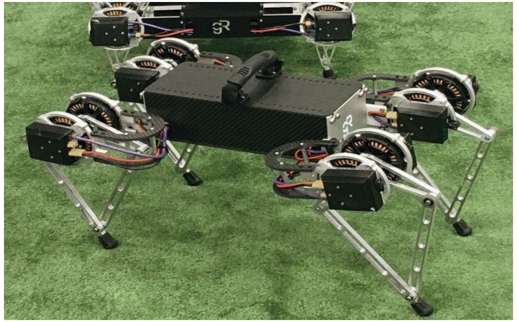
UNIVERSITY OF CAMBRIDGE

Google Brain Robotics: **Krzysztof Choromanski, Vikas Sindhwani**

University of Cambridge and Alan Turing Institute: **Mark Rowland**, **Richard E. Turner**, **Adrian Weller**

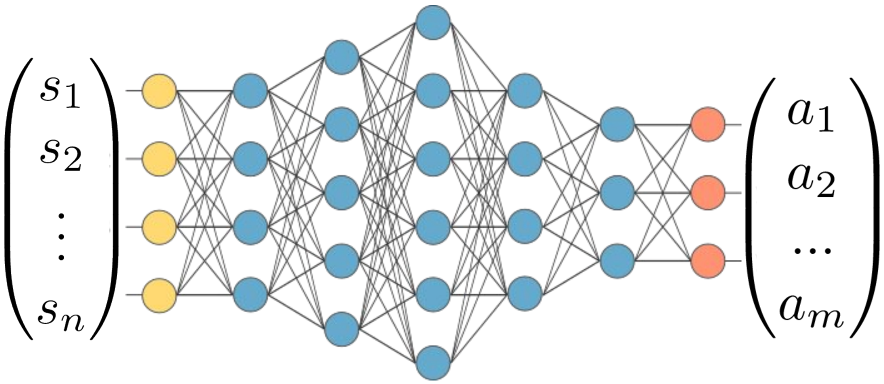




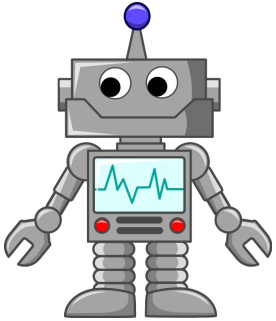


INNOVATION + ASSISTANCE

Gradient sensing - blackbox approach to RL

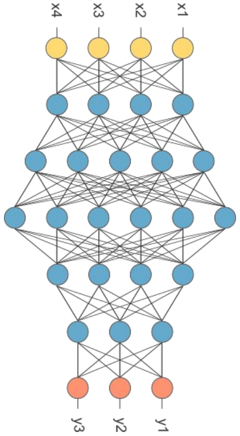


$$\pi : S \to A$$

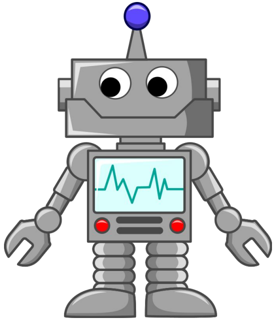


Gradient sensing - blackbox approach to RL

$$\begin{pmatrix} \theta_1 \\ \theta_2 \\ \vdots \\ \theta_L \end{pmatrix} \quad \boxed{F} \quad R_{\text{TOTAL}}$$



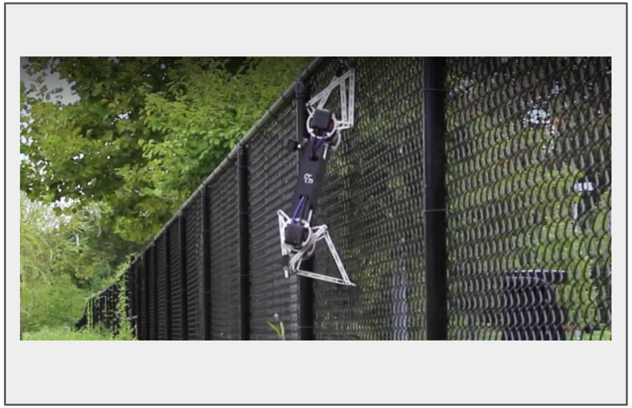
$$F : \mathbb{R}^d \rightarrow \mathbb{R}$$



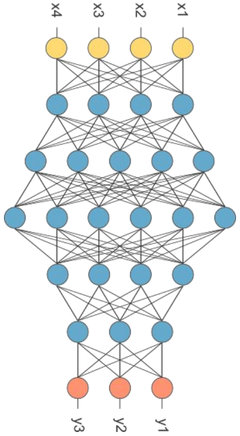
INNOVATION + ASSISTANCE

Gradient sensing - blackbox approach to RL

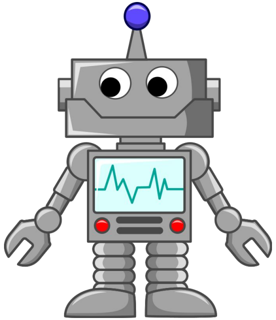
$$\begin{pmatrix}\theta_1\\ \theta_2\\ \vdots\\ \theta_L\end{pmatrix}$$



R_{TOTAL}



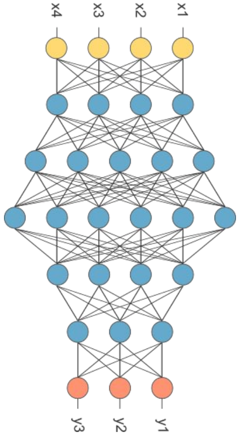
$$F : \mathbb{R}^d \rightarrow \mathbb{R}$$

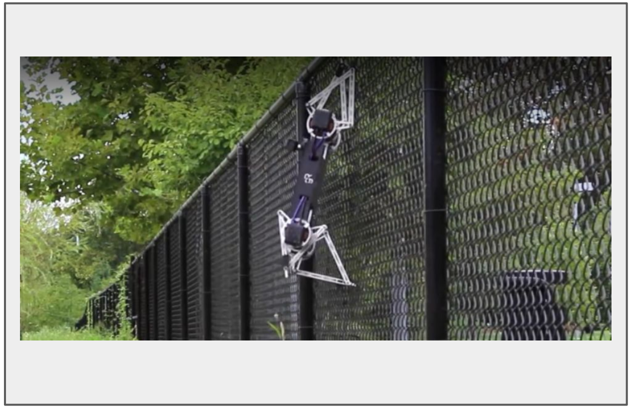


INNOVATION + ASSISTANCE

Gradient sensing - blackbox approach to RL

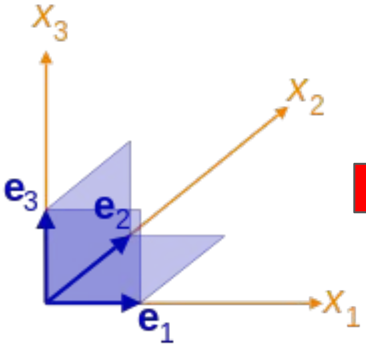
$$\begin{pmatrix}\theta_1\\ \theta_2\\ \vdots\\ \theta_L\end{pmatrix}$$



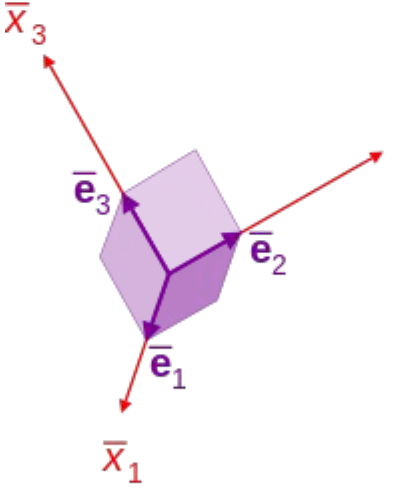


R_{TOTAL}

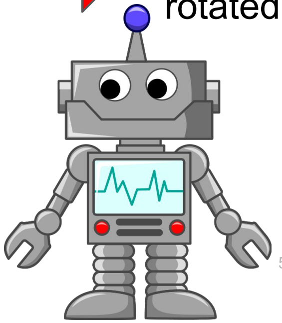
$$F:\mathbb{R}^d \rightarrow \mathbb{R}$$



finite difference gradient sensing

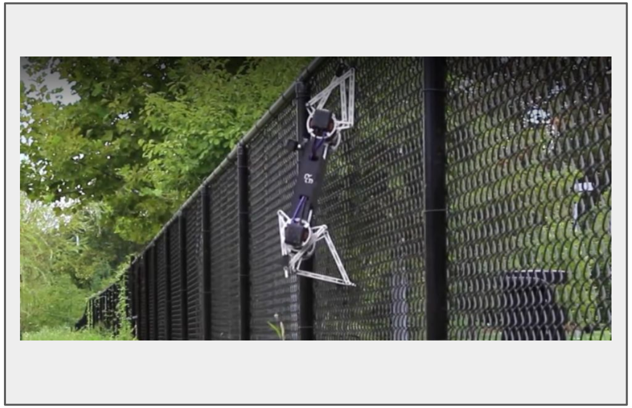


gradient sensing via randomly rotated basis



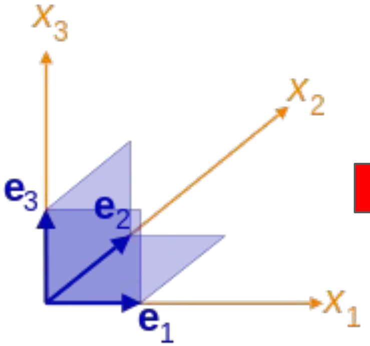
Gradient sensing - blackbox approach to RL

$$\begin{pmatrix} \theta_1 \\ \theta_2 \\ \vdots \\ \theta_L \end{pmatrix}$$

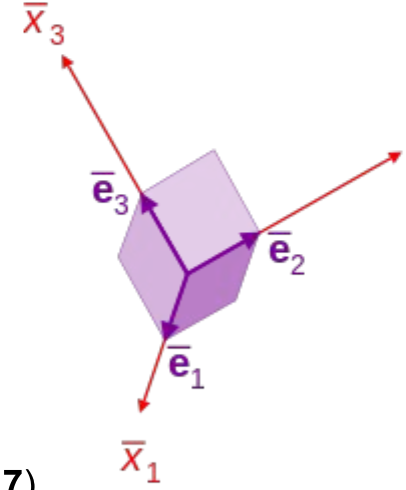


R_{TOTAL}

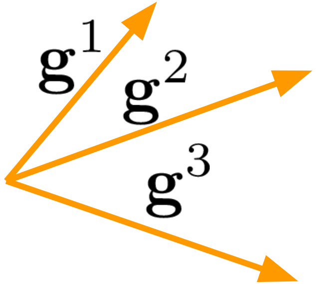
$$F : \mathbb{R}^d \rightarrow \mathbb{R}$$



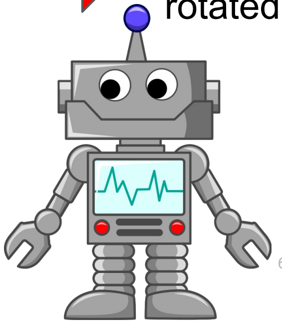
finite difference gradient sensing



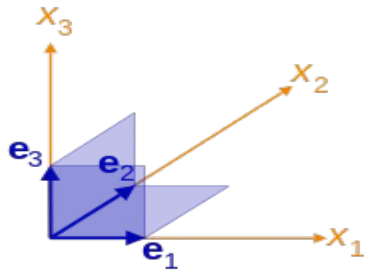
gradient sensing via randomly rotated basis



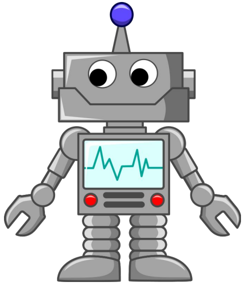
gradient sensing via independent gaussian vectors (**Salimans et al. 2017**)

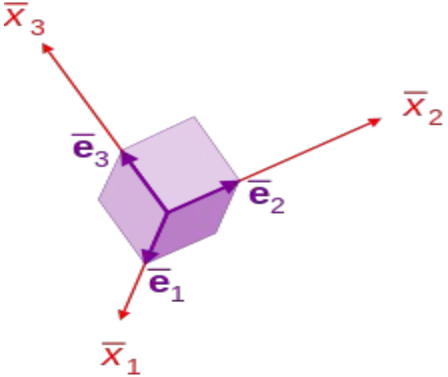


Gradient sensing - blackbox approach to RL

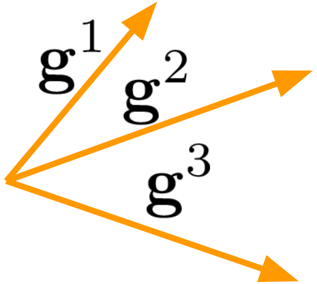


$$\mathbf{P} = \begin{pmatrix} 1 & 0 & \dots & 0 \\ 0 & 1 & \dots & 0 \\ \dots & \dots & \dots & \dots \\ 0 & 0 & \dots & 1 \end{pmatrix}$$





$$\mathbf{G}_{\mathrm{ort}} = \begin{pmatrix} g^{\mathrm{ort}}_{1,1} & g^{\mathrm{ort}}_{1,2} & \cdots & g^{\mathrm{ort}}_{1,n} \\ g^{\mathrm{ort}}_{2,1} & g^{\mathrm{ort}}_{2,2} & \cdots & g^{\mathrm{ort}}_{2,n} \\ \dots & \dots & \dots & \dots \\ g^{\mathrm{ort}}_{m,1} & g^{\mathrm{ort}}_{m,n} & \cdots & g^{\mathrm{ort}}_{m,n} \end{pmatrix}$$



$$\mathbf{G} = \begin{pmatrix} g_{1,1} & g_{1,2} & \cdots & g_{1,n} \\ g_{2,1} & g_{2,2} & \cdots & g_{2,n} \\ \dots & \dots & \dots & \dots \\ g_{m,1} & g_{m,n} & \cdots & g_{m,n} \end{pmatrix}$$

$$F : \mathbb{R}^d \rightarrow \mathbb{R}$$

Towards smooth relaxations

$$\max_{\mu \in \mathcal{P}(\mathbb{R}^d)} \mathbb{E}_{\theta \sim \mu}[F(\theta)]$$

family of probabilistic distributions on $\mathbb{R}^d$

Gaussian smoothings

$$\max_{\theta \in \mathbb{R}^d} J(\theta) = \mathbb{E}_{\phi \sim N(\theta, \sigma^2 I)}[F(\phi)]$$

- infinitely differentiable objective (**Nesterov & Spokoiny, 2017**)
- optimal value lower-bounds That of the original problem

Gradient sensing as a Monte-Carlo estimation

ES-style gradient estimator (Salimans et al. 2017):

$$\widehat{\nabla}_N J(\theta) = \frac{1}{N\sigma}\sum_{i=1}^{N} F(\theta + \sigma\epsilon_i)\epsilon_i$$

used in many evolutionary strategies papers, no control variate

Gradient estimator with antithetic pairs (Salimans et al. 2017):

$$\widehat{\nabla}_N J(\theta) = \frac{1}{2N\sigma}\sum_{i=1}^{N} (F(\theta + \sigma\epsilon_i)\epsilon_i - F(\theta - \sigma\epsilon_i)\epsilon_i)$$

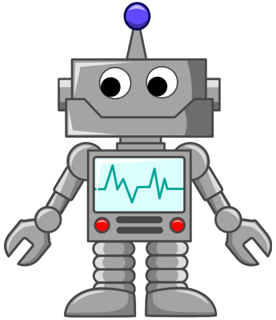
used for variance reduction

FD-style gradient estimator:

$$\widehat{\nabla}_{\boxed{N}} J(\theta) = \frac{1}{N\sigma}\sum_{i=1}^{N} (F(\theta + \sigma\epsilon_i)\epsilon_i - F(\theta)\epsilon_i)$$

number of samples

randomized version of a standard finite difference method



INNOVATION + ASSISTANCE

Gradient sensing as a Monte-Carlo estimation

ES-style gradient estimator (Salimans et al. 2017):

$$\widehat{\nabla}_N J(\theta) = \frac{1}{N\sigma} \sum_{i=1}^{N} F(\theta + \sigma\epsilon_i)\epsilon_i$$

used in many evolutionary strategies papers, no control variate

Gradient estimator with antithetic pairs (Salimans et al. 2017):

$$\widehat{\nabla}_N J(\theta) = \frac{1}{2N\sigma} \sum_{i=1}^{N} (F(\theta + \sigma\epsilon_i)\epsilon_i - F(\theta - \sigma\epsilon_i)\epsilon_i)$$

used for variance reduction

FD-style gradient estimator:

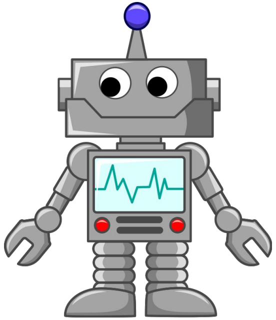
$$\widehat{\nabla}_{\boxed{N}} J(\theta) = \frac{1}{N\sigma} \sum_{i=1}^{N} (F(\theta + \sigma\epsilon_i)\epsilon_i - F(\theta)\epsilon_i)$$

number of samples

randomized version of a standard finite difference method

$$\widehat{\nabla}^{\mathrm{ort}}_N J(\theta)$$

$$\{\epsilon_1^{\mathrm{ort}}, \ldots, \epsilon_N^{\mathrm{ort}}\}$$



Gradient sensing as a Monte-Carlo estimation

ES-style gradient estimator (Salimans et al. 2017):

$$\widehat{\nabla}_N J(\theta) = \frac{1}{N\sigma}\sum_{i=1}^{N} F(\theta + \sigma\epsilon_i)\epsilon_i$$

used in many evolutionary strategies papers, no control variate

Gradient estimator with antithetic pairs (Salimans et al. 2017):

$$\widehat{\nabla}_N J(\theta) = \frac{1}{2N\sigma}\sum_{i=1}^{N} (F(\theta + \sigma\epsilon_i)\epsilon_i - F(\theta - \sigma\epsilon_i)\epsilon_i)$$

used for variance reduction

FD-style gradient estimator:

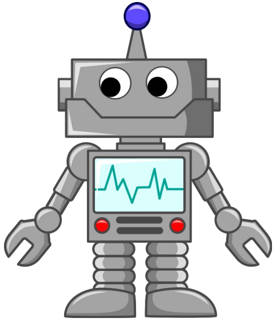
$$\widehat{\nabla}_{\boxed{N}} J(\theta) = \frac{1}{N\sigma}\sum_{i=1}^{N} (F(\theta + \sigma\epsilon_i)\epsilon_i - F(\theta)\epsilon_i)$$

number of samples

randomized version of a standard finite difference method

- all three estimators are unbiased
- empirically tested that none of them dominates the others

$$\widehat{\nabla}^{\mathrm{ort}}_N J(\theta) \rightarrow \{\epsilon_1^{\mathrm{ort}}, \ldots, \epsilon_N^{\mathrm{ort}}\}$$



Gradient sensing as a Monte-Carlo estimation

ES-style gradient estimator (Salimans et al. 2017):

$$\widehat{\nabla}_N J(\theta) = \frac{1}{N\sigma}\sum_{i=1}^{N} F(\theta + \sigma\epsilon_i)\epsilon_i$$

Gradient estimator with antithetic pairs (Salimans et al. 2017):

$$\widehat{\nabla}_N J(\theta) = \frac{1}{2N\sigma}\sum_{i=1}^{N} (F(\theta + \sigma\epsilon_i)\epsilon_i - \boxed{F(\theta - \sigma\epsilon_i)\epsilon_i})$$

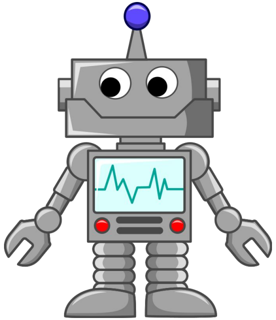
FD-style gradient estimator:

$$\widehat{\nabla}_{\boxed{N}} J(\theta) = \frac{1}{N\sigma}\sum_{i=1}^{N} (F(\theta + \sigma\epsilon_i)\epsilon_i - \boxed{F(\theta)\epsilon_i})$$

number of samples

- all three estimators are unbiased
- empirically tested that none of them dominates the others
- how to learn optimal control variates ?

$$\{\epsilon_1^{\mathrm{ort}}, \ldots, \epsilon_N^{\mathrm{ort}}\}$$



Gradient sensing as a Monte-Carlo estimation

ES-style gradient estimator (Salimans et al. 2017):

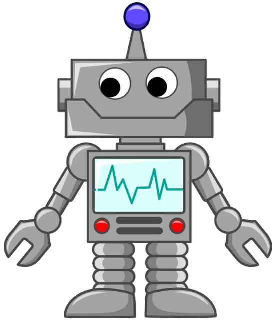
$$\widehat{\nabla}_N J(\theta) = \frac{1}{N\sigma}\sum_{i=1}^{N} F(\theta + \sigma\epsilon_i)\epsilon_i$$

Gradient estimator with antithetic pairs (Salimans et al. 2017):

$$\widehat{\nabla}_N J(\theta) = \frac{1}{2N\sigma}\sum_{i=1}^{N} (F(\theta + \sigma\epsilon_i)\epsilon_i - F(\theta - \sigma\epsilon_i)\epsilon_i)$$

FD-style gradient estimator:

$$\widehat{\nabla}_N J(\theta) = \frac{1}{N\sigma}\sum_{i=1}^{N} (F(\theta + \sigma\epsilon_i)\epsilon_i - F(\theta)\epsilon_i)$$



Some recent success stories of structured random estimators (very biased selection !):

- “Initialization matters: Orthogonal Predictive State Recurrent Neural Networks”, *Choromanski, Downey, Boots* (to appear at **ICLR’18**)
- “Optimizing Simulations with Noise-Tolerant Structured Exploration”, *Choromanski, Iscen, Sindhwani, Tan, Coumans* (to appear at **ICRA’18**)
- “The Geometry of Random Features”, *Choromanski, Rowland, Sarlos, Sindhwani, Turner, Weller* (to appear at **AISTATS’18**)
- “The Unreasonable Effectiveness of Structured Random Orthogonal Embeddings”, *Choromanski, Rowland, Weller* (**NIPS’17**)
- “Structured adaptive and random spinners for fast machine learning computations”, *Bojarski, Choromanska, Choromanski, Fagan, Gouy-Pailler, Morvan, Sakr, Sarlos, Atif* (**AISTATS’17**)
- “Orthogonal Random Features”, *Yu, Suresh, Choromanski, Holtmann-Rice, Kumar* (**NIPS’16**)
- “Recycling Randomness with Structure for Sublinear time Kernel Expansion”, *Choromanski, Sindhwani* (**ICML’16**)
- “Binary embeddings with structured hashed projections”, *Choromanska, Choromanski, Bojarski, Jebara, Kumar, LeCun* (**ICML’16**)

Gradient sensing as a Monte-Carlo estimation

ES-style gradient estimator (Salimans et al. 2017):

$$\widehat{\nabla}_N J(\theta) = \frac{1}{N\sigma}\sum_{i=1}^{N} F(\theta + \sigma\epsilon_i)\epsilon_i$$

Gradient estimator with antithetic pairs (Salimans et al. 2017):

$$\widehat{\nabla}_N J(\theta) = \frac{1}{2N\sigma}\sum_{i=1}^{N} (F(\theta + \sigma\epsilon_i)\epsilon_i - F(\theta - \sigma\epsilon_i)\epsilon_i)$$

FD-style gradient estimator:

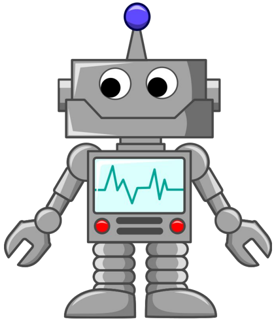
$$\widehat{\nabla}_N J(\theta) = \frac{1}{N\sigma}\sum_{i=1}^{N} (F(\theta + \sigma\epsilon_i)\epsilon_i - F(\theta)\epsilon_i)$$

Theorem (Choromanski, Rowland, Sindhwani, Turner, Weller'18)

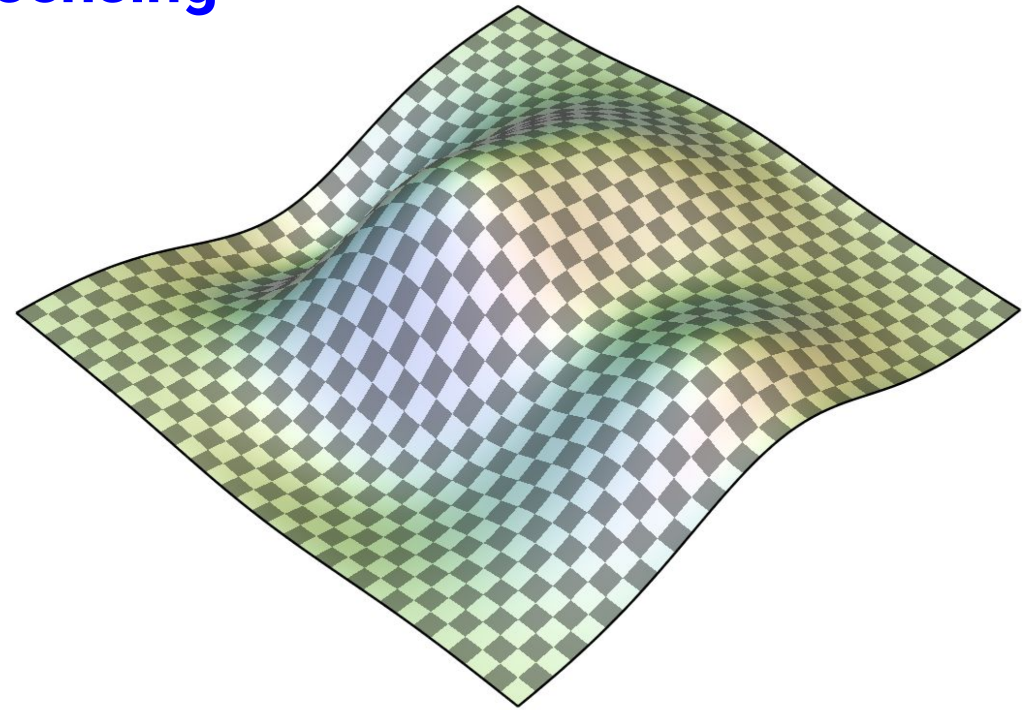
The orthogonal gradient estimator $\widehat{\nabla}_N^{\mathrm{ort}} J(\theta)$ is unbiased and yields lower MSE than the unstructured estimator $\widehat{\nabla}_N J(\theta)$, namely:

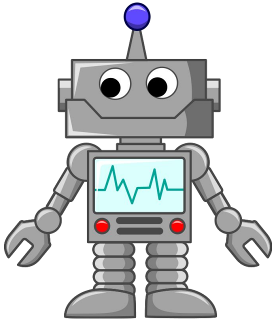
$$\mathrm{MSE}(\widehat{\nabla}_N^{\mathrm{ort}} J(\theta)) =$$

$$\mathrm{MSE}(\widehat{\nabla}_N J(\theta)) - \frac{N-1}{N}\|\nabla J(\theta)\|_2^2.$$



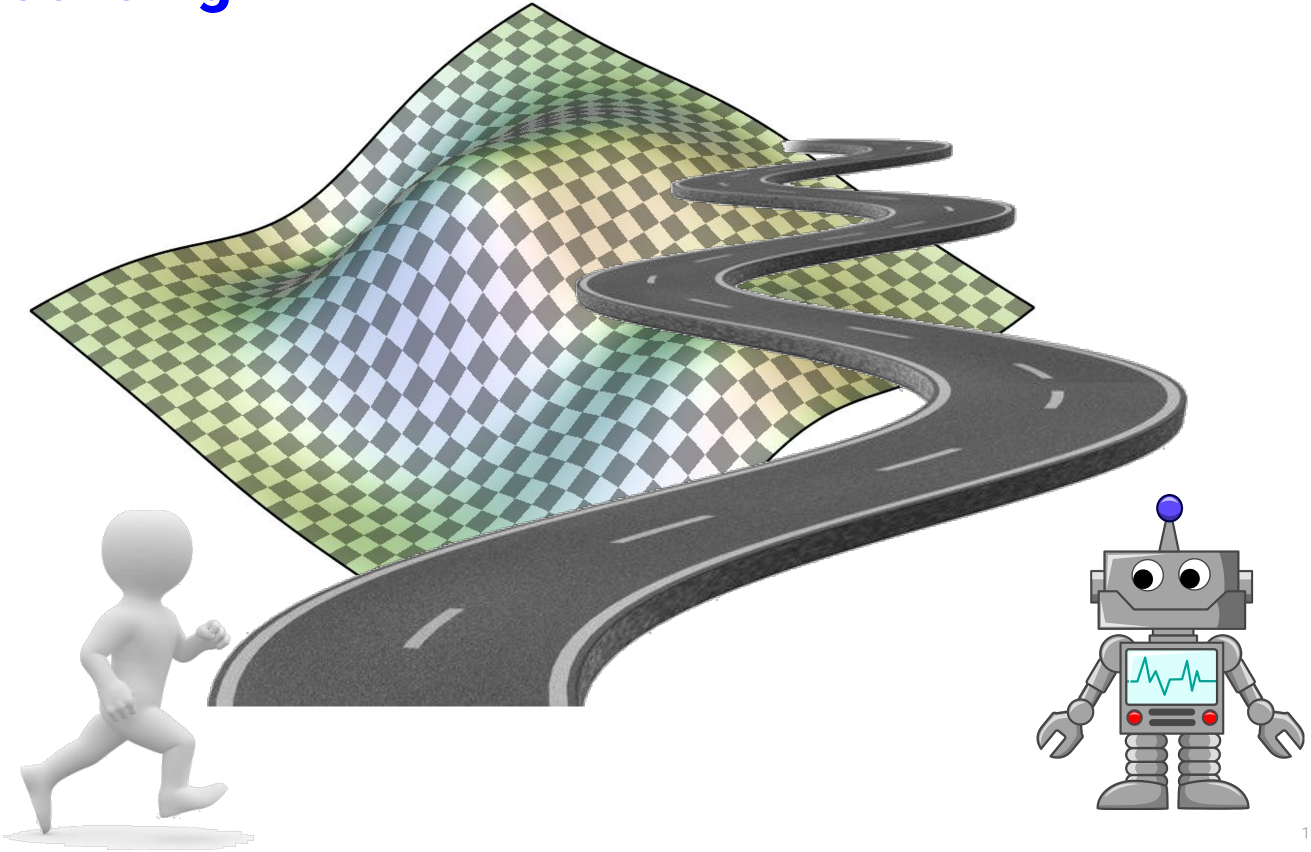
Efficiency of the orthogonal space exploration for gradient sensing





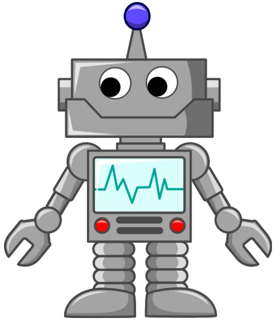
INNOVATION + ASSISTANCE

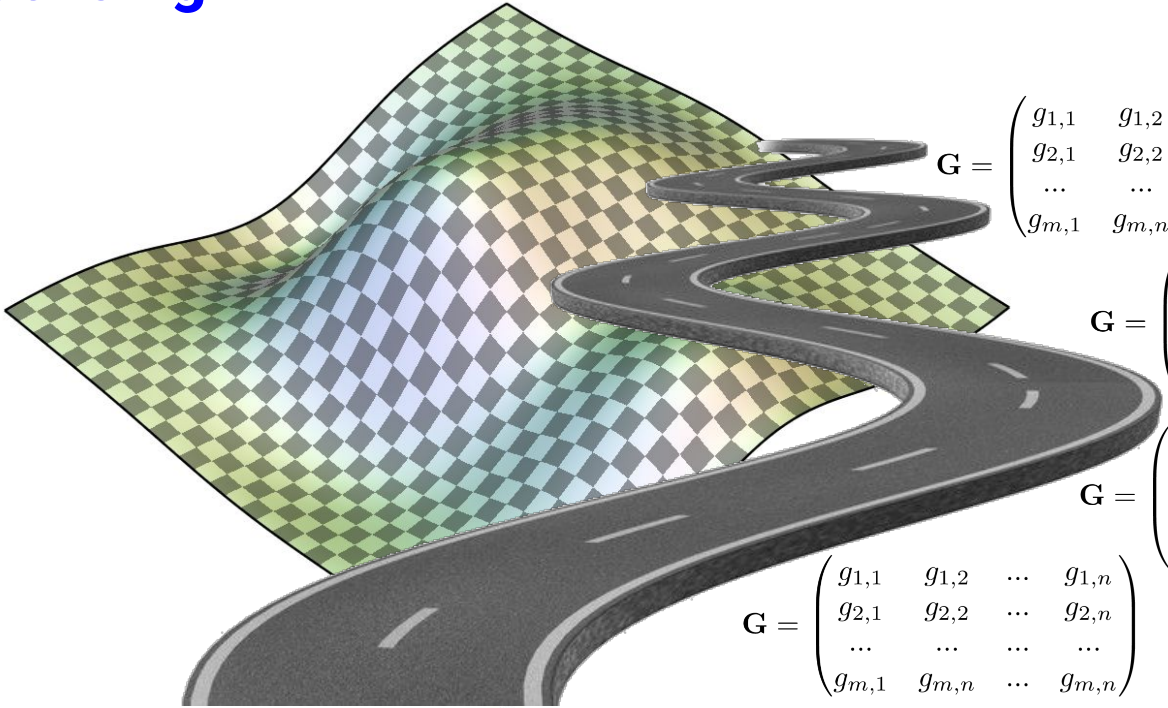
Efficiency of the orthogonal space exploration for gradient sensing



INNOVATION + ASSISTANCE

Efficiency of the orthogonal space exploration for gradient sensing





$$\mathbf{G} = \begin{pmatrix} g_{1,1} & g_{1,2} & \cdots & g_{1,n} \\ g_{2,1} & g_{2,2} & \cdots & g_{2,n} \\ \cdots & \cdots & \cdots & \cdots \\ g_{m,1} & g_{m,n} & \cdots & g_{m,n} \end{pmatrix}$$

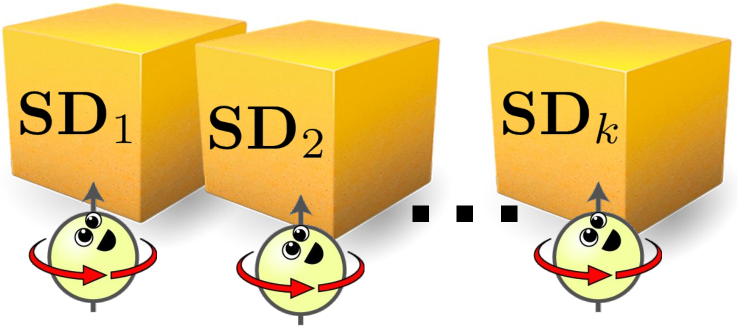
$$\mathbf{G} = \begin{pmatrix} g_{1,1} & g_{1,2} & \cdots & g_{1,n} \\ g_{2,1} & g_{2,2} & \cdots & g_{2,n} \\ \cdots & \cdots & \cdots & \cdots \\ g_{m,1} & g_{m,n} & \cdots & g_{m,n} \end{pmatrix}$$

$$\mathbf{G} = \begin{pmatrix} g_{1,1} & g_{1,2} & \cdots & g_{1,n} \\ g_{2,1} & g_{2,2} & \cdots & g_{2,n} \\ \cdots & \cdots & \cdots & \cdots \\ g_{m,1} & g_{m,n} & \cdots & g_{m,n} \end{pmatrix}$$

$$\mathbf{G} = \begin{pmatrix} g_{1,1} & g_{1,2} & \cdots & g_{1,n} \\ g_{2,1} & g_{2,2} & \cdots & g_{2,n} \\ \cdots & \cdots & \cdots & \cdots \\ g_{m,1} & g_{m,n} & \cdots & g_{m,n} \end{pmatrix}$$

$$\mathbf{G} = \begin{pmatrix} g_{1,1} & g_{1,2} & \cdots & g_{1,n} \\ g_{2,1} & g_{2,2} & \cdots & g_{2,n} \\ \cdots & \cdots & \cdots & \cdots \\ g_{m,1} & g_{m,n} & \cdots & g_{m,n} \end{pmatrix}$$

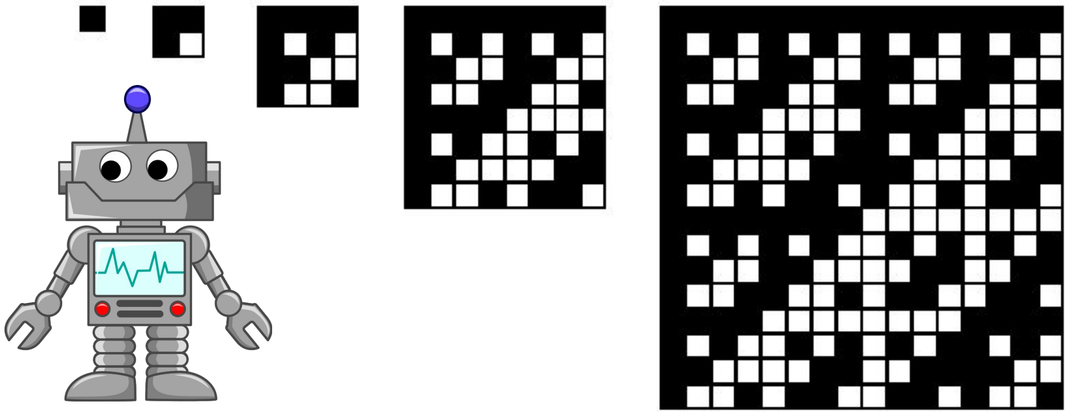
Efficiency of the orthogonal space exploration for gradient sensing - discrete space sensing



$$\mathbf{M}^{(k)}_{\mathbf{S}\mathcal{R}} = \prod_{i=1}^{k} \mathbf{SD}_i^{\boxed{(\mathcal{R})}} \rightarrow |\lambda_i| = 1$$

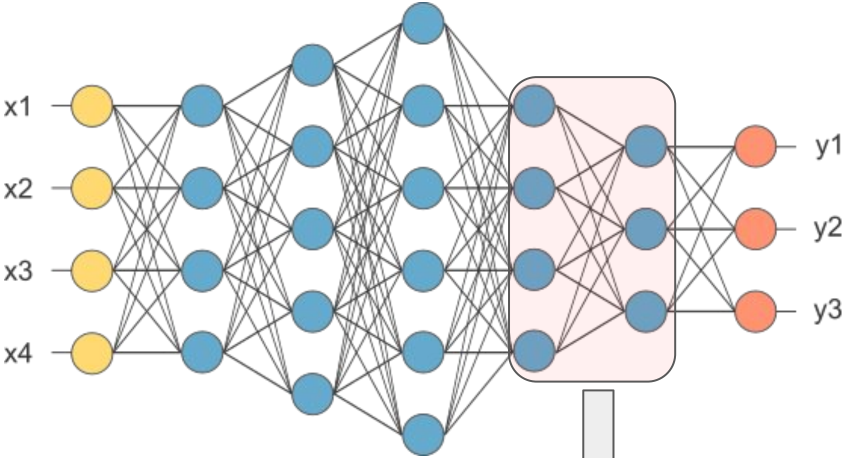
$$\lambda_i \sim Unif\{-1,+1\}$$

$\mathbf{S}_0\ \mathbf{S}_1\ \mathbf{S}_2\ \mathbf{S}_3\ \mathbf{S}_4$

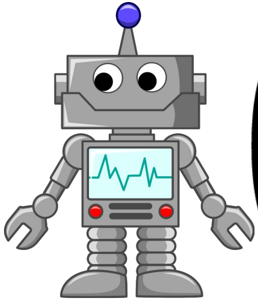


$$\mathbf{D} = \begin{pmatrix} \lambda_1 & 0 & 0 & \cdots \\ 0 & \lambda_2 & 0 & \cdots \\ 0 & 0 & \lambda_3 & \cdots \\ \vdots & \vdots & \vdots & \ddots \end{pmatrix}$$

Efficiency of the orthogonal space exploration for gradient sensing - structured neural networks



$$\begin{pmatrix} m_1 & m_2 & m_3 & m_4 \\ m_5 & m_1 & m_2 & m_3 \\ m_6 & m_5 & m_1 & m_2 \end{pmatrix}$$

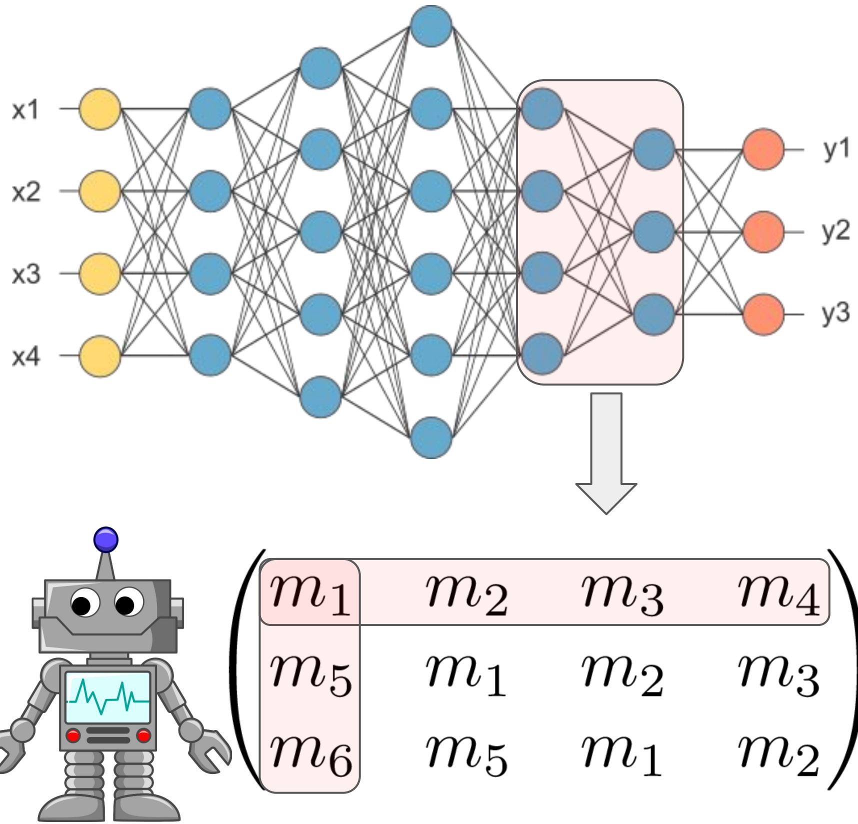


Structured Matrices: Toeplitz

$$\begin{bmatrix} t_0 & t_{-1} & \cdots & t_{-(n-1)} \\ t_1 & t_0 & \cdots & \vdots \\ \vdots & \vdots & \vdots & t_{-1} \\ t_{n-1} & \cdots & t_1 & t_0 \end{bmatrix}$$

- $n \times n$ matrix parameterized by $2n - 1$ numbers: constant diagonal values
- $O(n \log n)$ matrix-vector products, Linear Systems
- Applications
  - Implements One-dimensional Linear Convolutions
  - Arises naturally in time series analysis and dynamical systems.
  - Related matrix: Hankel - antidiagonals are constant

Efficiency of the orthogonal space exploration for gradient sensing - structured neural networks



$$\begin{pmatrix} m_1 & m_2 & m_3 & m_4 \\ m_5 & m_1 & m_2 & m_3 \\ m_6 & m_5 & m_1 & m_2 \end{pmatrix}$$

Sylvester Displacement and Unit-Circulants

- The *Sylvester* displacement operator is defined by,

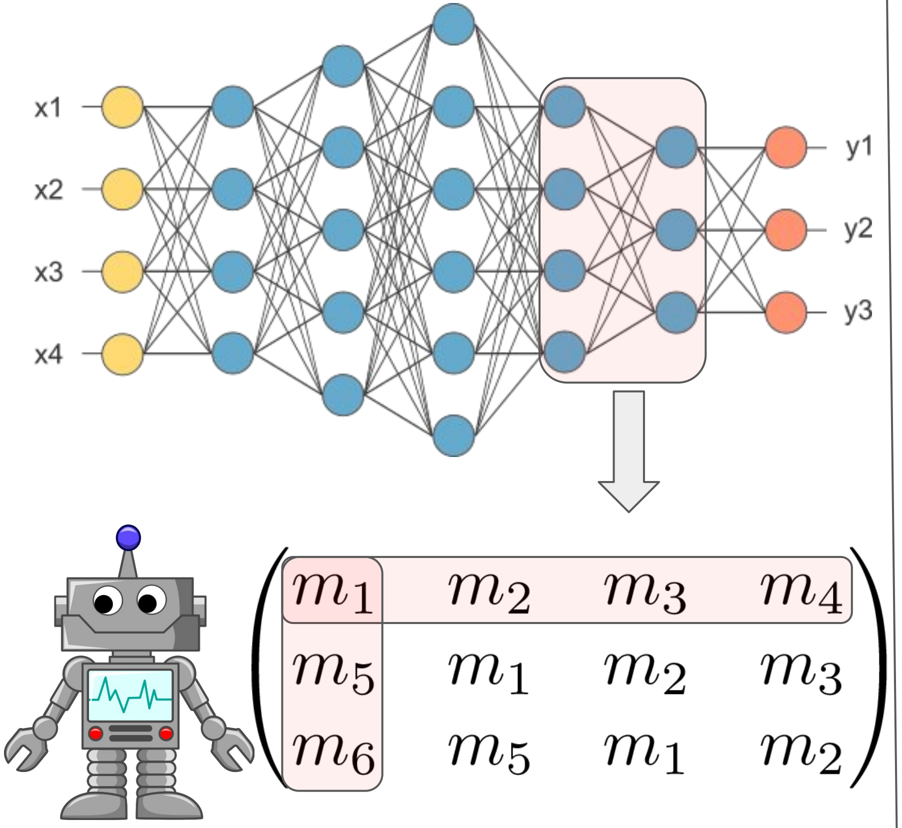
$$\nabla_{\mathbf{A},\mathbf{B}}[\mathbf{M}] = \mathbf{AM} - \mathbf{MB}$$

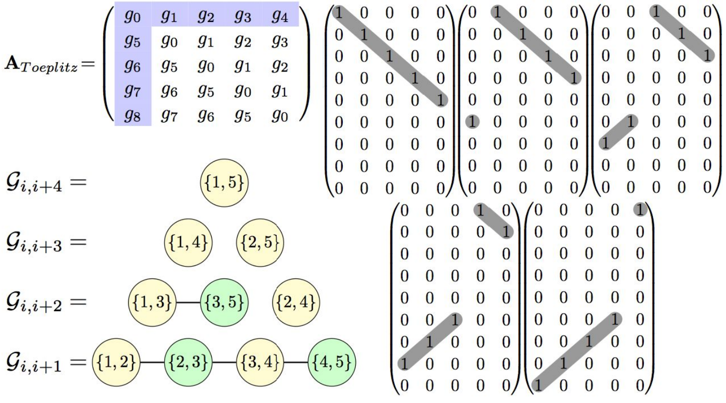
  where $\mathbf{A} \in \mathbb{R}^{m\times m}, \mathbf{B} \in \mathbb{R}^{n\times n}$ are fixed *operator matrices*.
- Design displacement operators by carefully choosing $\mathbf{A}, \mathbf{B}$.
- Shift-and-Scale matrices are called f-unit Circulant matrices:

$$\mathbf{Z}_f = \begin{bmatrix} 0 & 0 & \dots & f \\ 1 & 0 & \dots & 0 \\ \vdots & \vdots & \vdots & \vdots \\ 0 & \dots & 1 & 0 \end{bmatrix}, \quad \mathbf{Z}_f \begin{pmatrix} v_1 \\ v_2 \\ v_3 \\ \vdots \\ v_n \end{pmatrix} = \begin{pmatrix} f v_n \\ v_1 \\ v_2 \\ \vdots \\ v_{n-1} \end{pmatrix}$$

  - $\mathbf{Z}_f^n = f\mathbf{I}$
  - Upshifts with $\mathbf{Z}_f^T$
  - $\mathbf{Z}_f^{-1} = \mathbf{Z}_{f^{-1}}^T$

Efficiency of the orthogonal space exploration for gradient sensing - structured neural networks



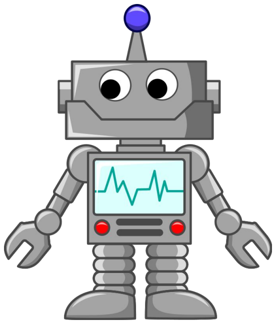


Very Low-displacement Rank Property

Structured Matrix $\mathbf{M}$	$\mathbf{A}$	$\mathbf{B}$	rank($\nabla_{\mathbf{A},\mathbf{B}}[\mathbf{M}]$)
Toeplitz and its inverse	$\mathbf{Z}_1$	$\mathbf{Z}_{-1}$	≤ 2
Hankel and its inverse	$\mathbf{Z}_1$	$\mathbf{Z}_0^T$	≤ 2
Toeplitz + Hankel	$\mathbf{Z}_0 + \mathbf{Z}_0^T$	$\mathbf{Z}_0 + \mathbf{Z}_0^T$	≤ 4
Vandermonde $V(\mathbf{v})$	$diag(\mathbf{v})$	$\mathbf{Z}_0$	≤ 1
Inverse of Vandermonde i.e. $V(\mathbf{v})^{-1}$	$\mathbf{Z}_0$	$diag(\mathbf{v})$	≤ 1
Transpose of Vandermonde i.e. $V(\mathbf{v})^T$	$\mathbf{Z}_0^T$	$diag(\mathbf{v})$	≤ 1
Cauchy $\mathbf{C}(\mathbf{s},\mathbf{t})$	$diag(\mathbf{s})$	$diag(\mathbf{t})$	≤ 1
Inverse of Cauchy i.e. $\mathbf{C}(\mathbf{s},\mathbf{t})^{-1}$	$diag(\mathbf{t})$	$diag(\mathbf{s})$	≤ 1

Experimental results: orthogonal blackbox gradient sensing for low-dimensional problems

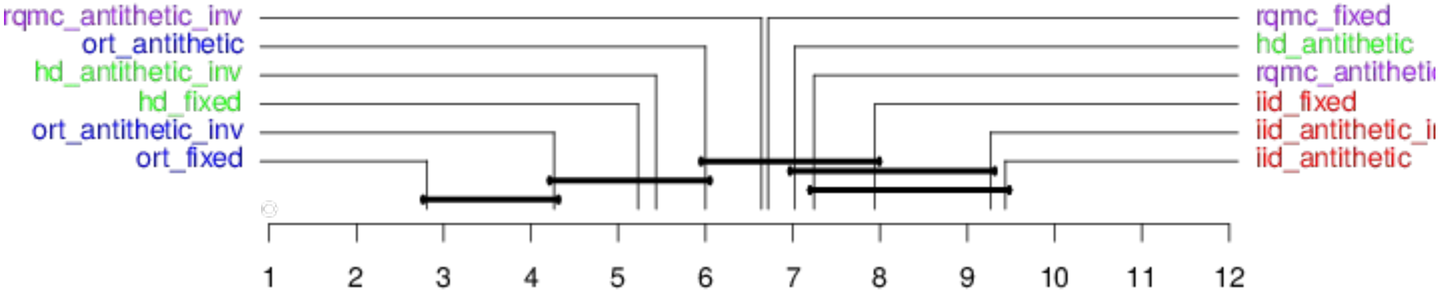
DFO benchmarking suite, More & Wild (2009)



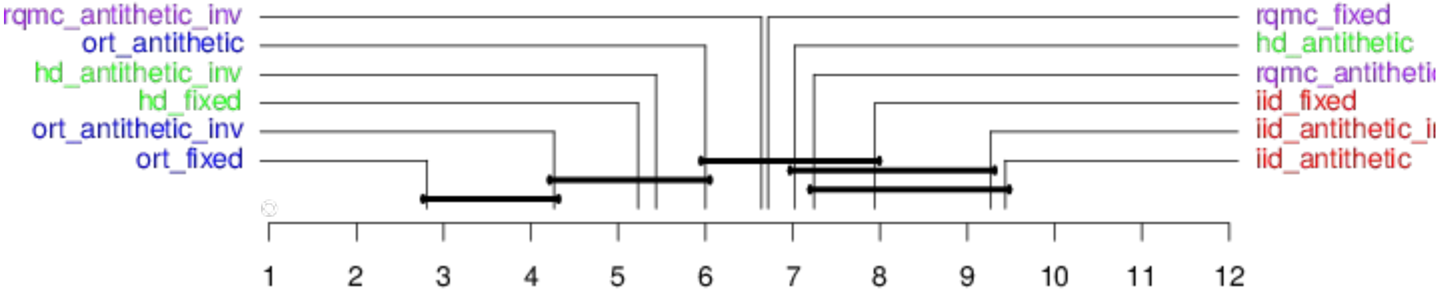
53 different optimization problems, four settings:

- stochastic
- deterministic noise
- smooth problems
- non-differentiable problems

- we compute average ranking of the methods against each other in terms of quality of final objective value in each optimisation task
- we then compare these average rankings using multiple hypothesis testing as described in **Demsar (2006)**



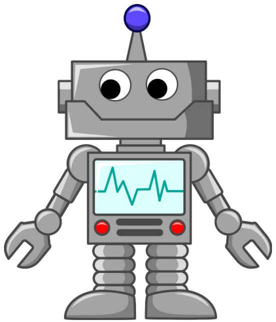
stochastic noise



deterministic noise

Experimental results: orthogonal blackbox gradient sensing for low-dimensional problems

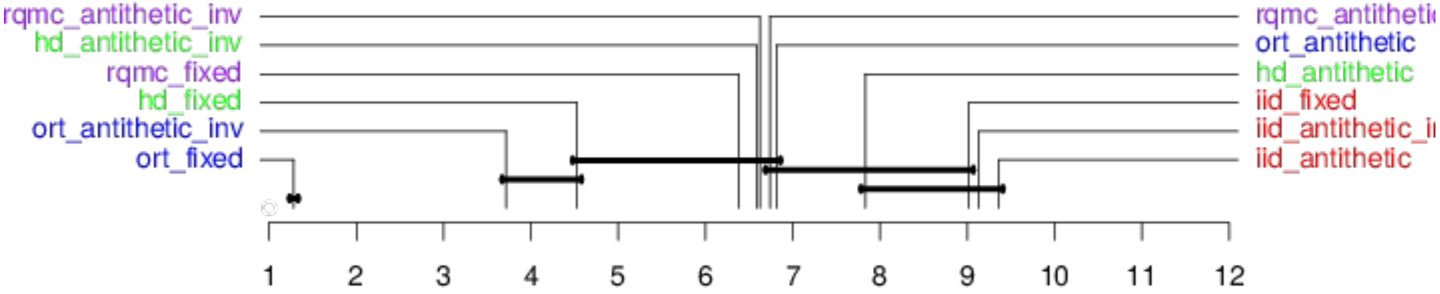
DFO benchmarking suite, More & Wild (2009)



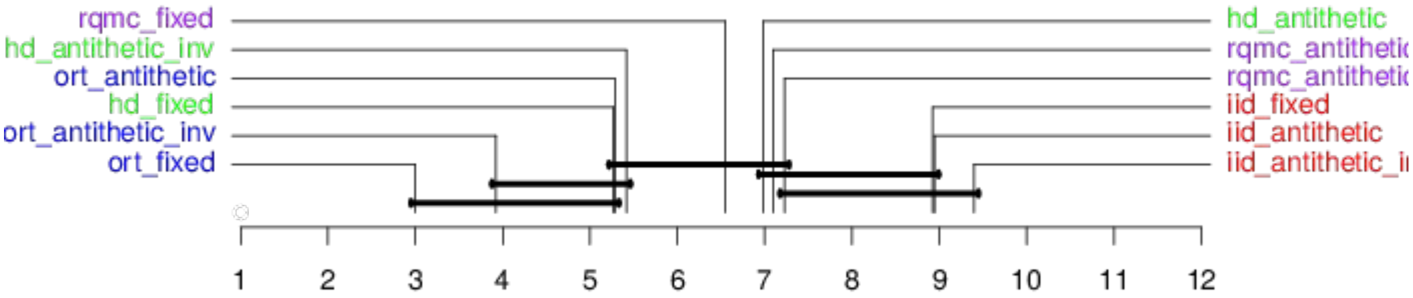
53 different optimization problems, four settings:

- stochastic
- deterministic noise
- smooth problems
- non-differentiable problems

- we compute average ranking of the methods against each other in terms of quality of final objective value in each optimisation task
- we then compare these average rankings using multiple hypothesis testing as described in **Demsar (2006)**

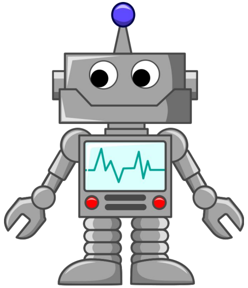


smooth problems



nondifferentiable problems

Experimental results: learning compressed policies for RL tasks



Optimization setting:

- AdamOptimizer with $\alpha = 0.01$ and $\sigma = 0.02$
- no heuristics used (e.g no fitness shaping)
- Base compressed setting: two hidden layers of size **h=41**

Tested variants:

- structured architectures + structured exploration (**STST**)
- structured architectures + unstructured exploration (**STUN**)
- unstructured architectures + unstructured exploration (**UNUN**)

Structured space exploration strategies:

- Gaussian orthogonal matrices
- matrices HD (k=1); 256- or 512-dimensional parameter vectors

solved by STST

Tested environments:

- Swimmer
- Ant
- HalfCheetah
- Hopper
- Humanoid
- Walker2d
- Pusher
- Reacher
- Striker
- Thrower
- ContMountainCar
- Pendulum
- Minitaur walking

Distributed TF training on at most **400** machines

Experimental results: learning compressed policies for RL tasks

Optimization setting:

- AdamOptimizer with $\alpha = 0.01$ and $\sigma = 0.02$
- no heuristics used (e.g no fitness shaping)
- Base compressed setting: two hidden layers of size **h=41**, **Toeplitz** matrices

Tested variants:

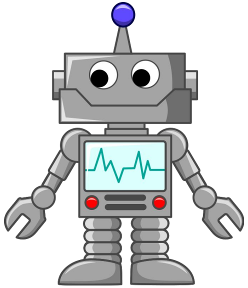
Learns reward **1842**

- structured architectures + structured exploration (**STST**)
- structured architectures + unstructured exploration (**STUN**)
- unstructured architectures + unstructured exploration (**UNUN**)

Structured space exploration strategies:

- Gaussian orthogonal matrices
- matrices HD (**k=1**); **256**- or **512**-dimensional parameter vectors

solved by STST



Tested environments:

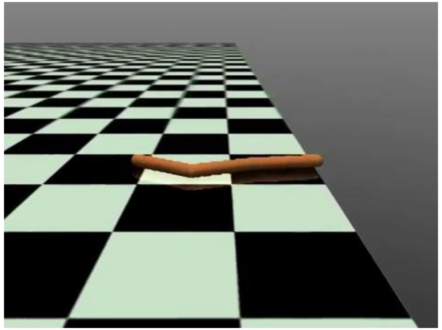
- Swimmer
- Ant
- HalfCheetah
- Hopper
- Humanoid
- Walker2d
- Pusher
- Reacher
- Striker
- Thrower
- ContMountainCar
- Pendulum
- Minitaur walking

	GaussOrt	Hadamard	baseline
Swimmer	253	253	1408
Ant	362	254	4896
HalfCheetah	266	254	2174
Hopper	257	254	1536
Humanoid	636	510	13664
Walker2d	266	254	1824
Pusher	273	255	2048
Reacher	256	256	1189
Striker	273	255	2048
Thrower	273	255	2048
ContMountCar	246	246	1184
Pendulum	247	247	1216
Minitaur	279	256	2240

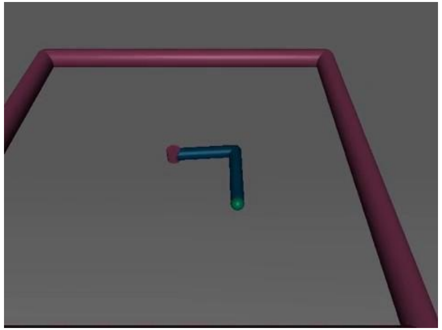
Distributed TF training on at most **400** machines

279-dimensional neural network learns reward **4.83** for rollouts of length **500**

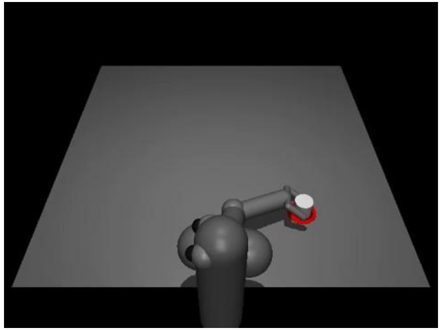
Experimental results: learning compressed policies for RL tasks



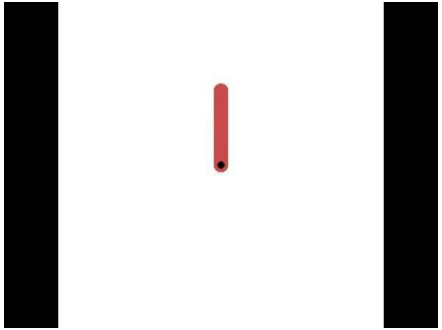
d=253



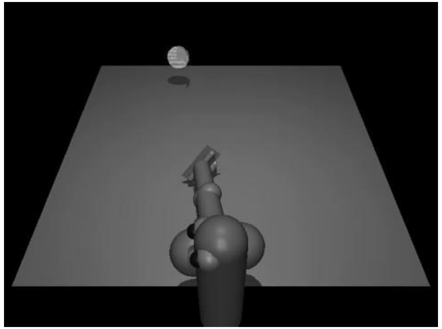
d=256



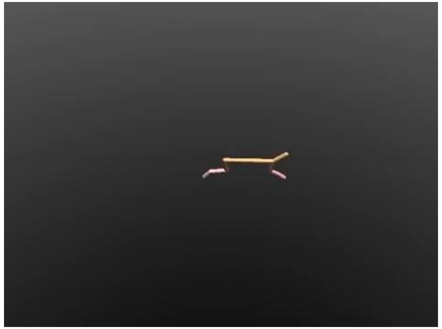
d=273



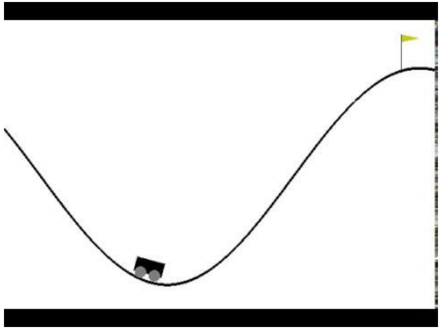
d=247



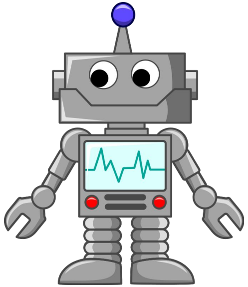
d=273



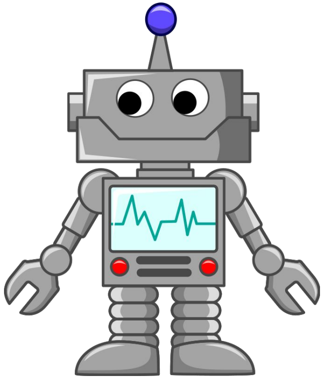
d=266

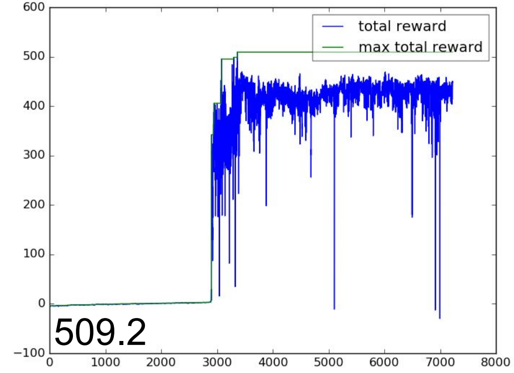


d=246

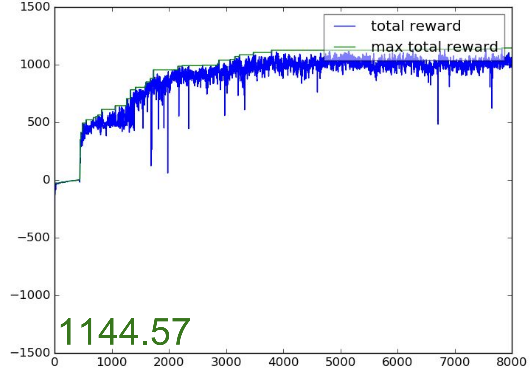


Experimental results: learning curves - Ant

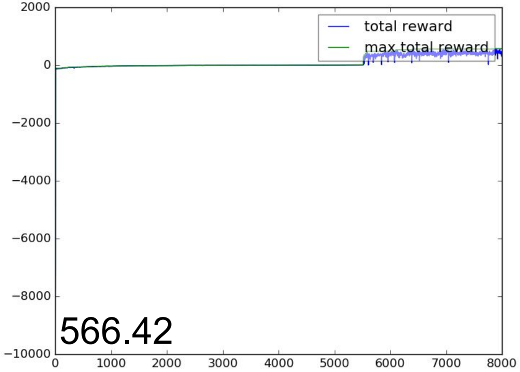




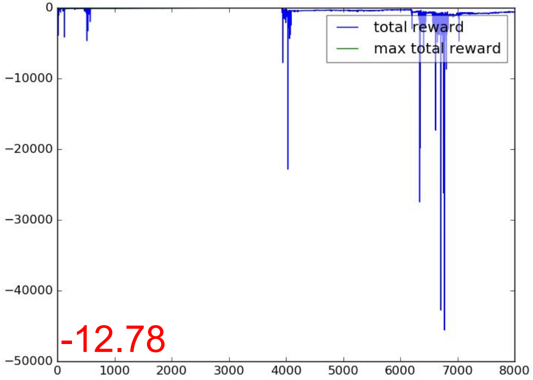
STST(GaussOrt)



STST(Hadamard)



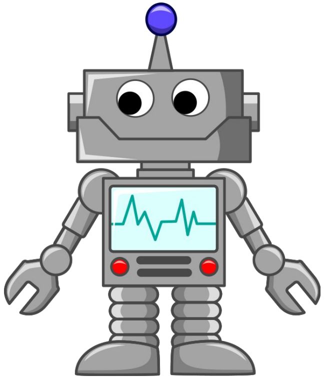
STUN

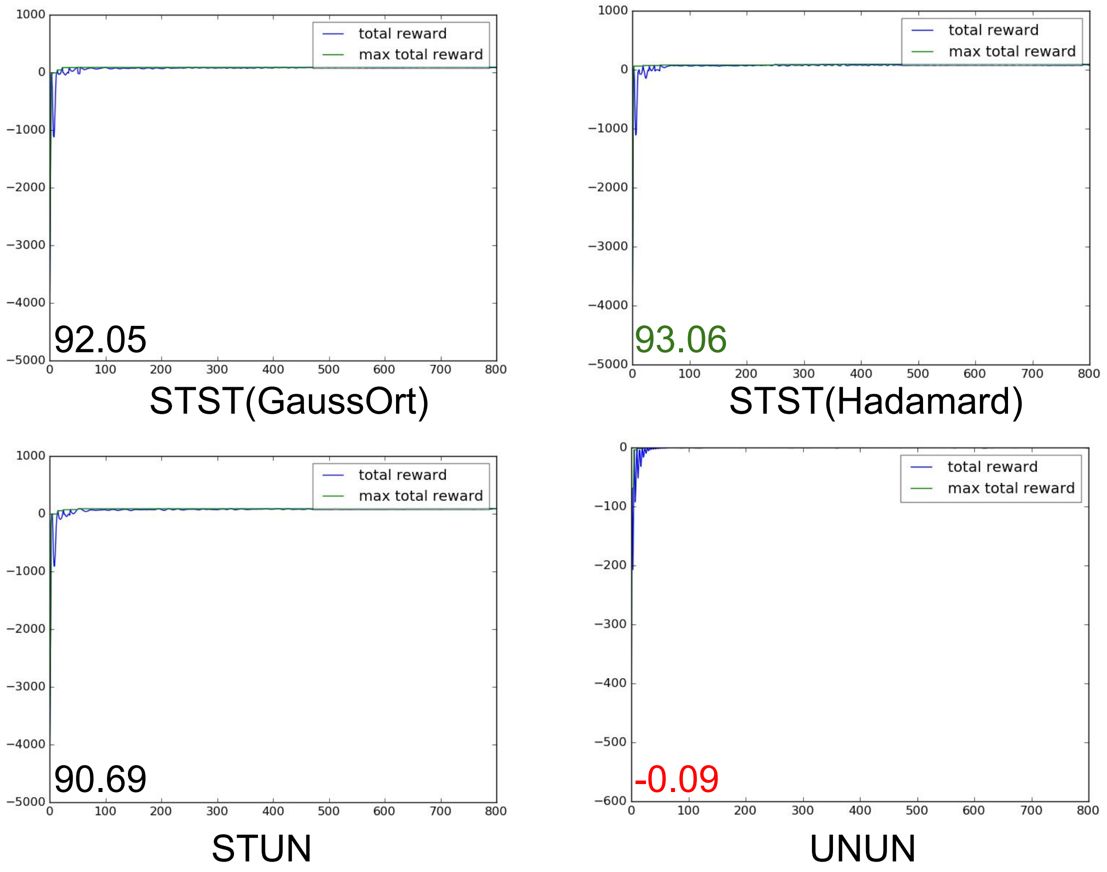


UNUN

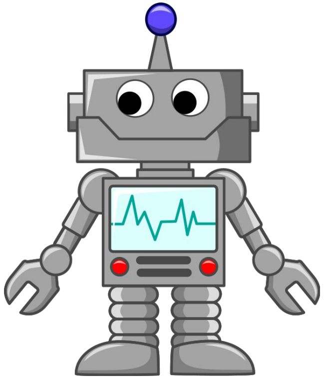
INNOVATION + ASSISTANCE

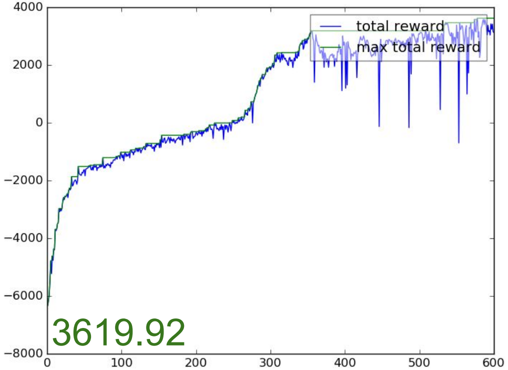
Experimental results: learning curves - Cont. Mountain Car



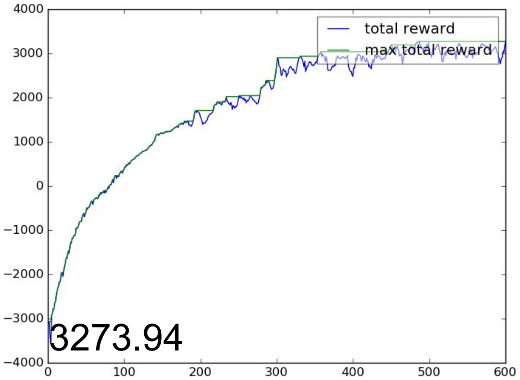


Experimental results: learning curves - HalfCheetah

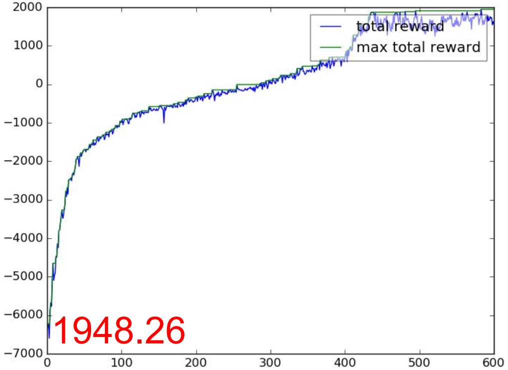




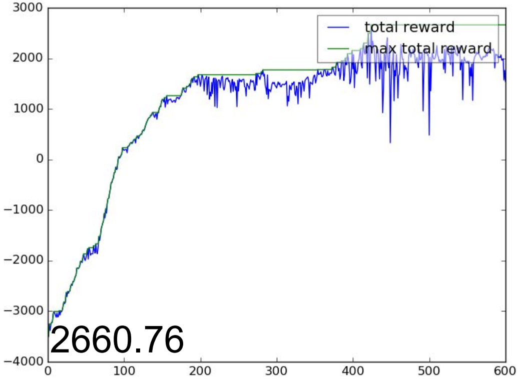
STST(GaussOrt)



STST(Hadamard)

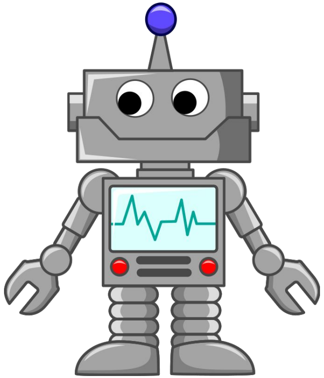


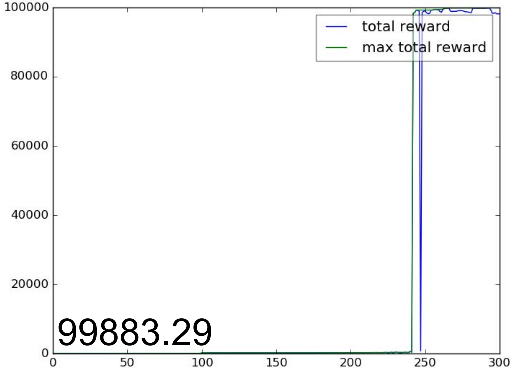
STUN



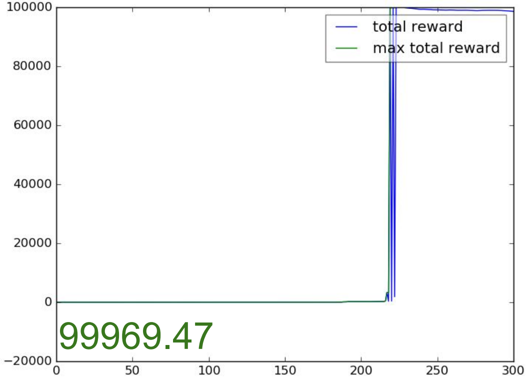
UNUN

Experimental results: learning curves - Hopper

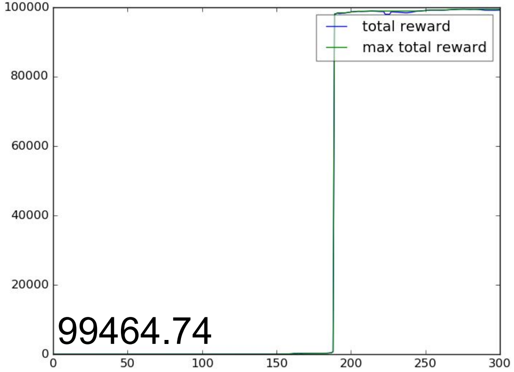




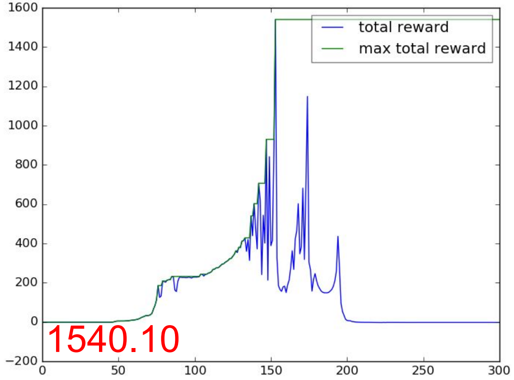
STST(GaussOrt)



STST(Hadamard)

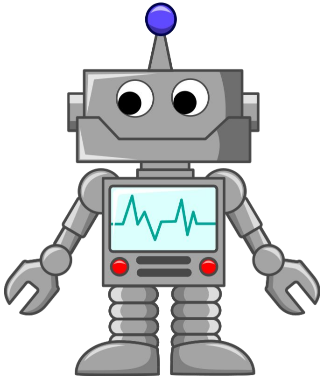


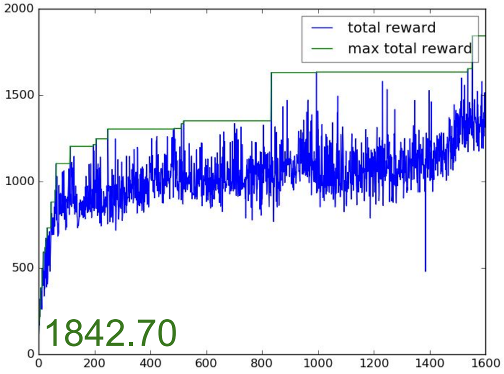
STUN



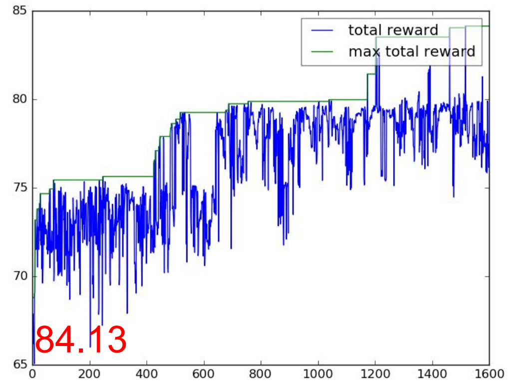
UNUN

Experimental results: learning curves - Humanoid

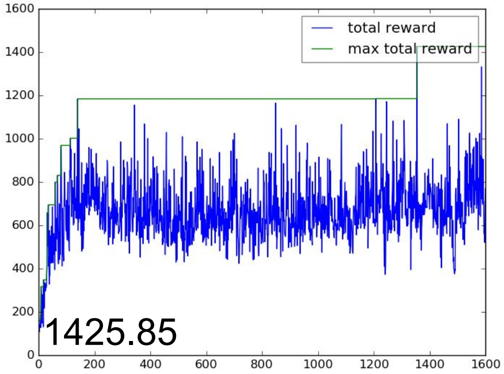




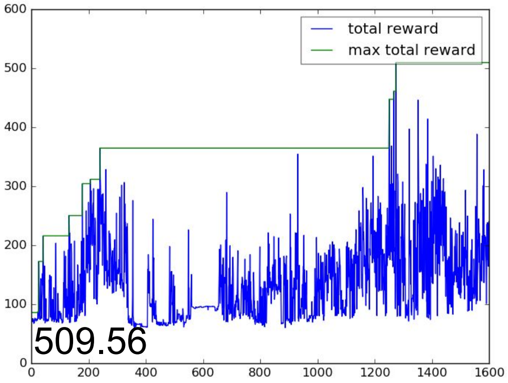
STST(GaussOrt)



STST(Hadamard)

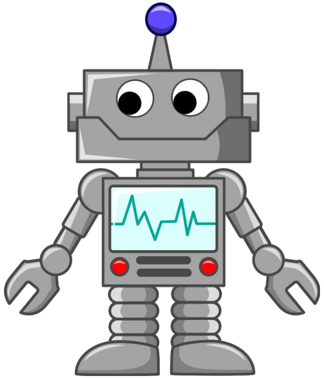


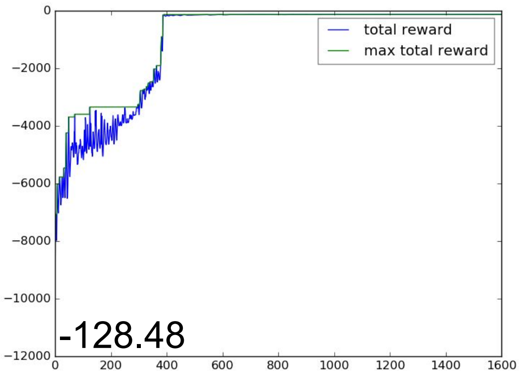
STUN



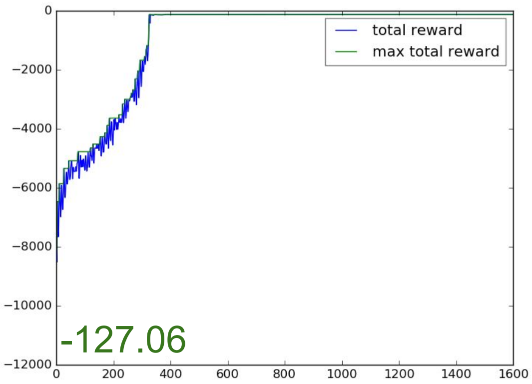
UNUN

Experimental results: learning curves - Pendulum

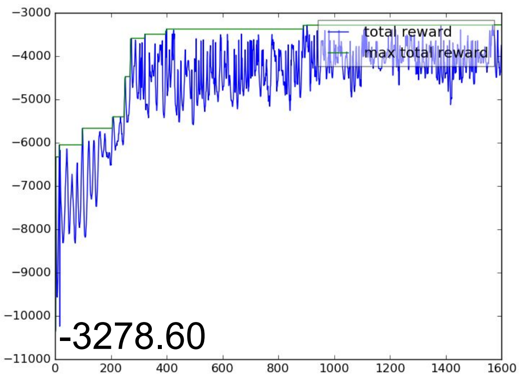




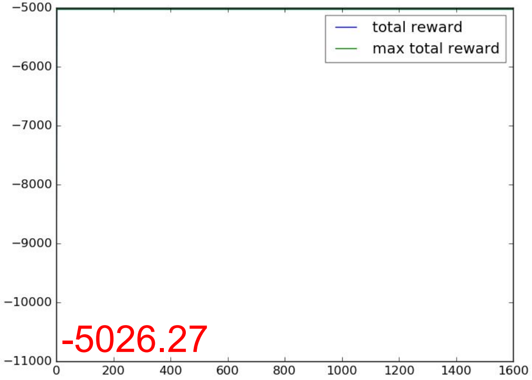
STST(GaussOrt)



STST(Hadamard)

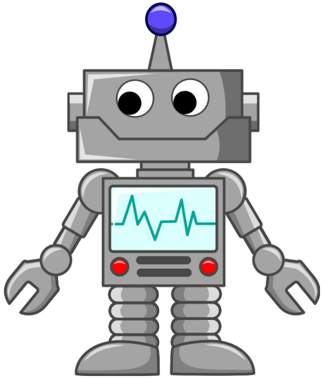


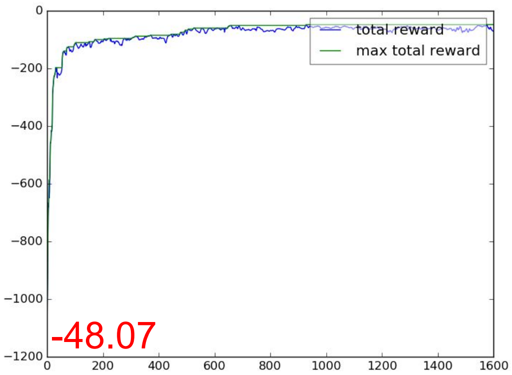
STUN



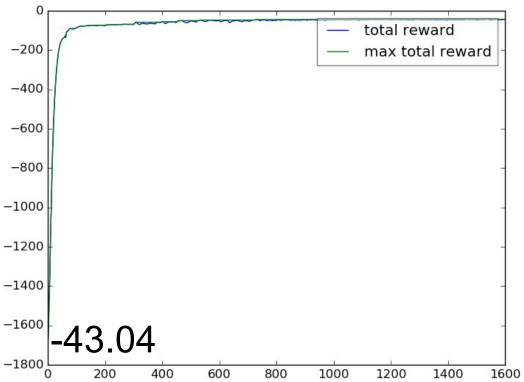
UNUN

Experimental results: learning curves - Pusher

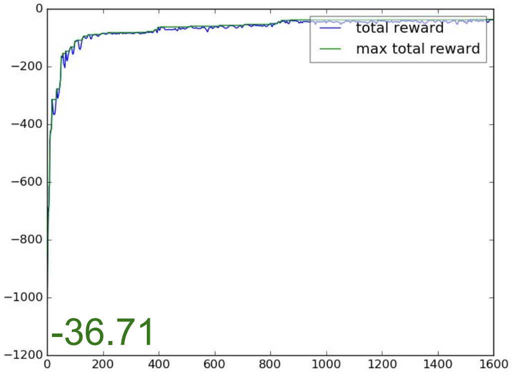




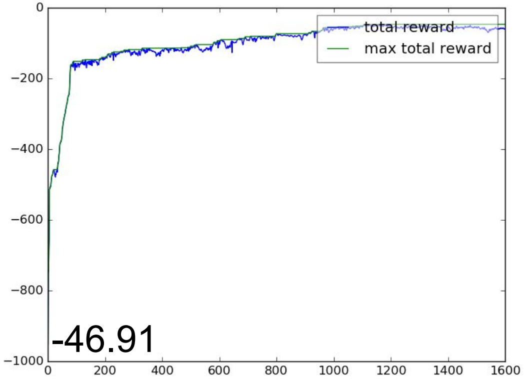
STST(GaussOrt)



STST(Hadamard)

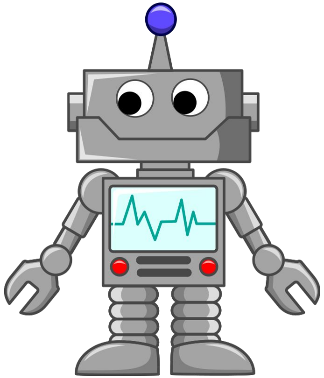


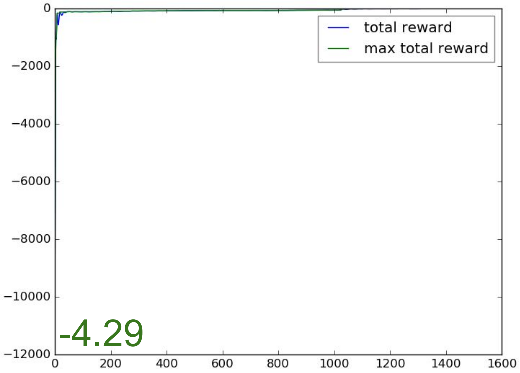
STUN



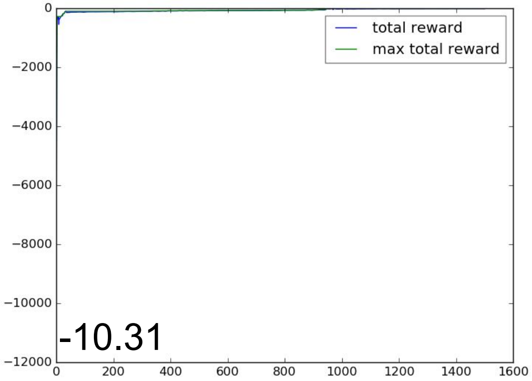
UNUN

Experimental results: learning curves - Reacher

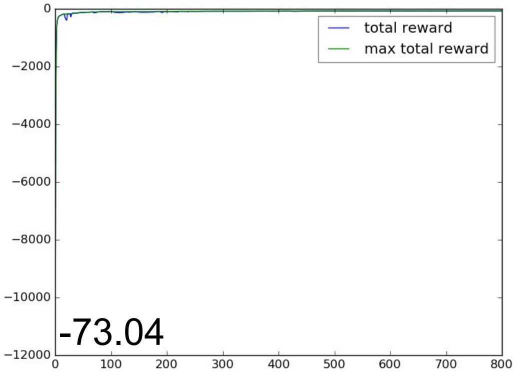




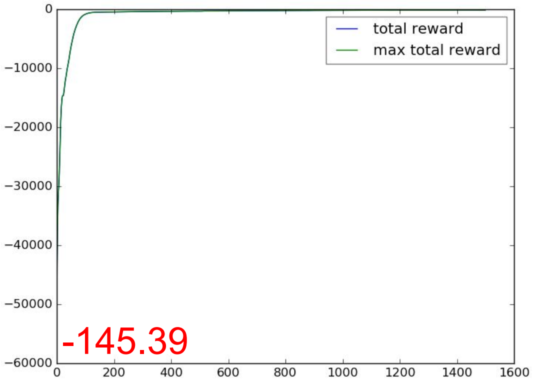
STST(GaussOrt)



STST(Hadamard)

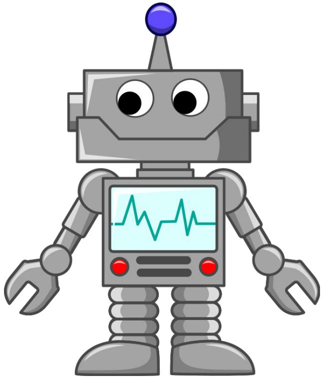


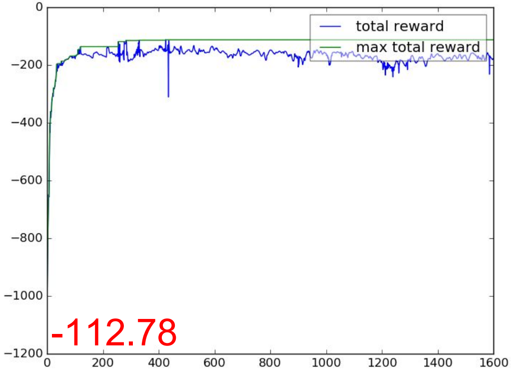
STUN



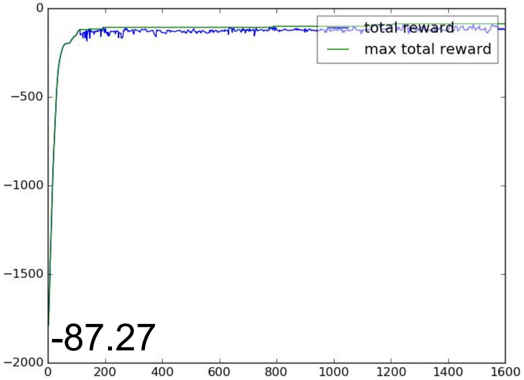
UNUN

Experimental results: learning curves - Striker

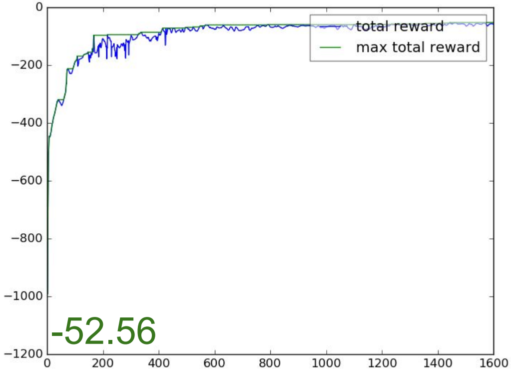




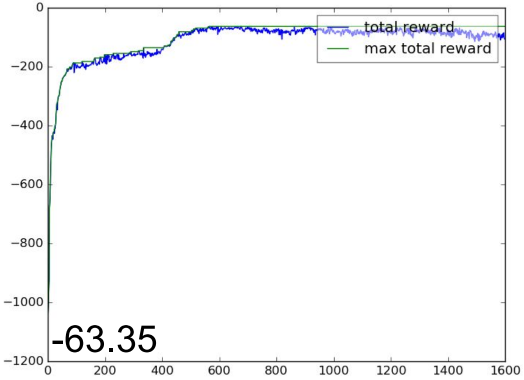
STST(GaussOrt)



STST(Hadamard)

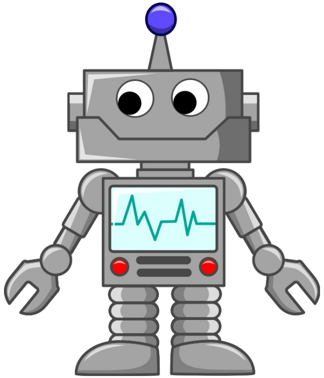


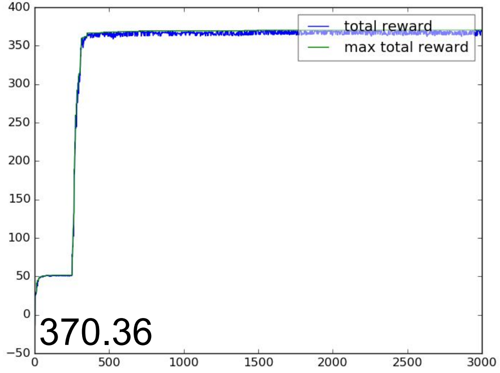
STUN



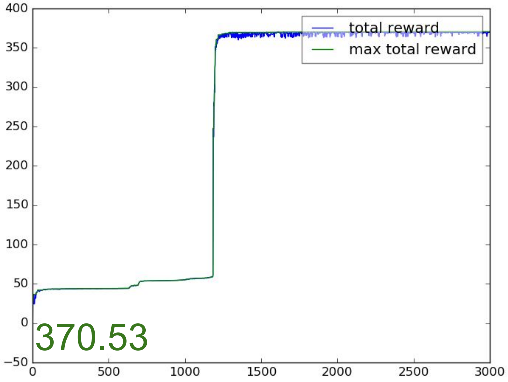
UNUN

Experimental results: learning curves - Swimmer

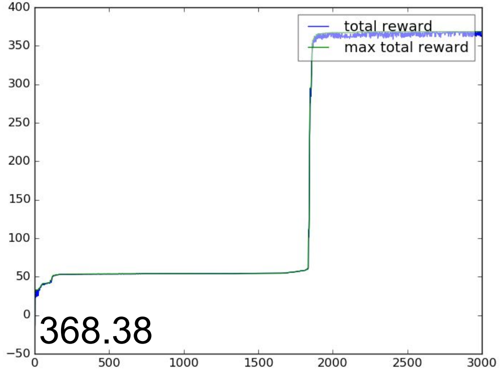




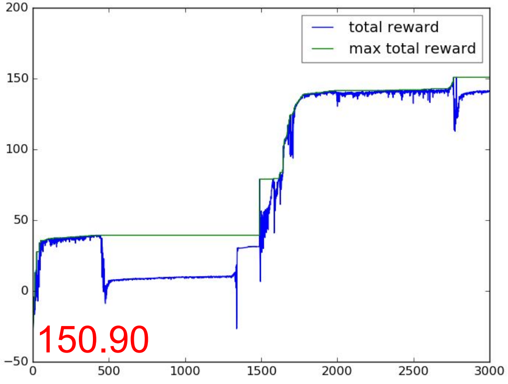
STST(GaussOrt)



STST(Hadamard)



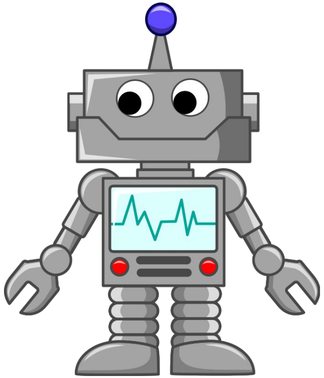
STUN

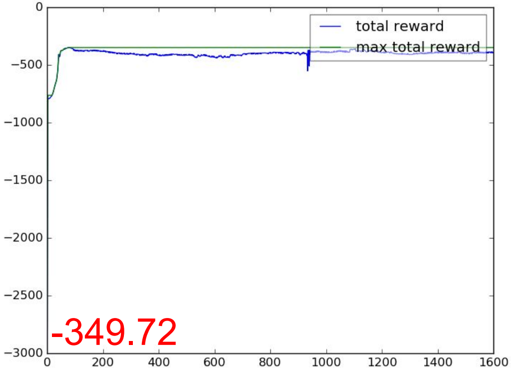


UNUN

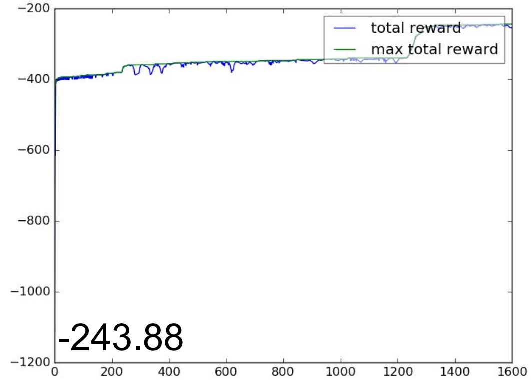
INNOVATION + ASSISTANCE

Experimental results: learning curves - Thrower

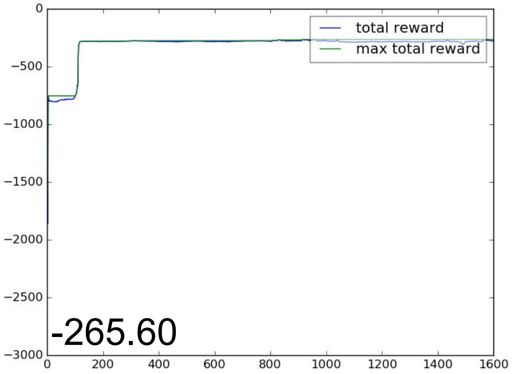




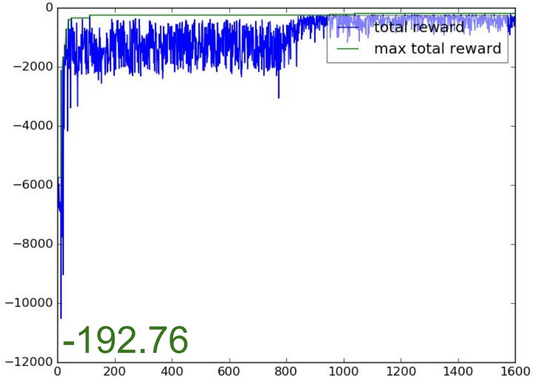
STST(GaussOrt)



STST(Hadamard)

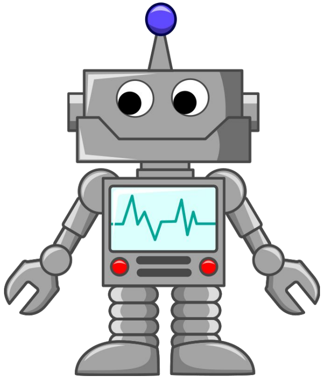


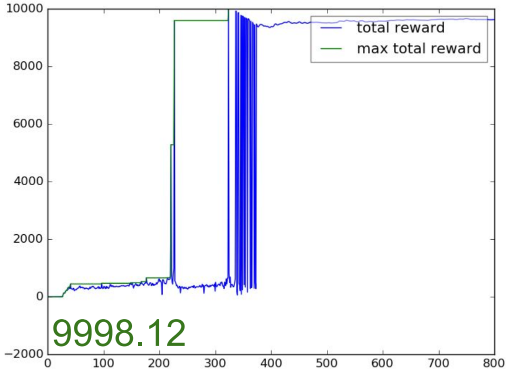
STUN



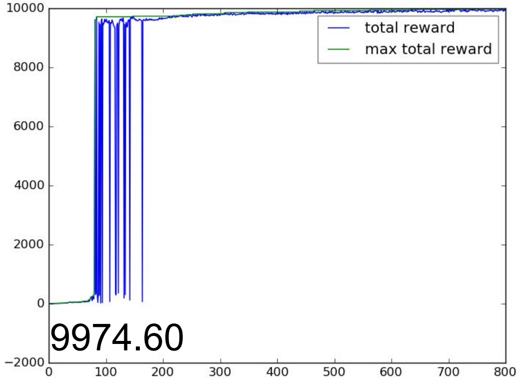
UNUN

Experimental results: learning curves - Walker2d

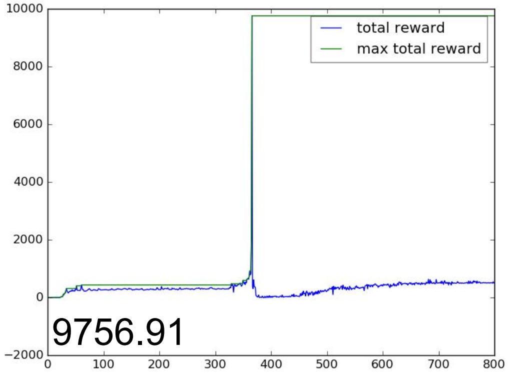




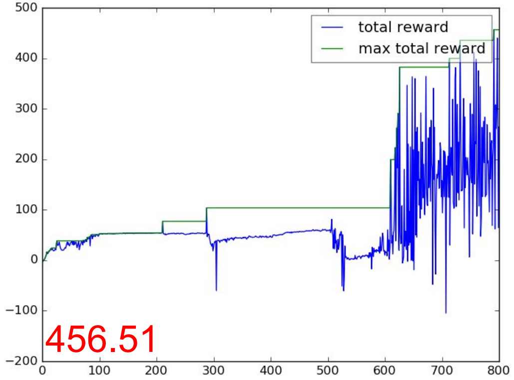
STST(GaussOrt)



STST(Hadamard)



STUN



UNUN

Success stories - teaching “Smoky” to walk (ICRA’18)

joint work with: Atil Iscen, Vikas Sindhwani, Jie Tan and Erwin Coumans