bit.ly/rpf_nips
@ianosband

Randomized Prior Functions for Deep Reinforcement Learning

Ian Osband, John Aslanides, Albin Cassirer

DeepMind

Reinforcement Learning

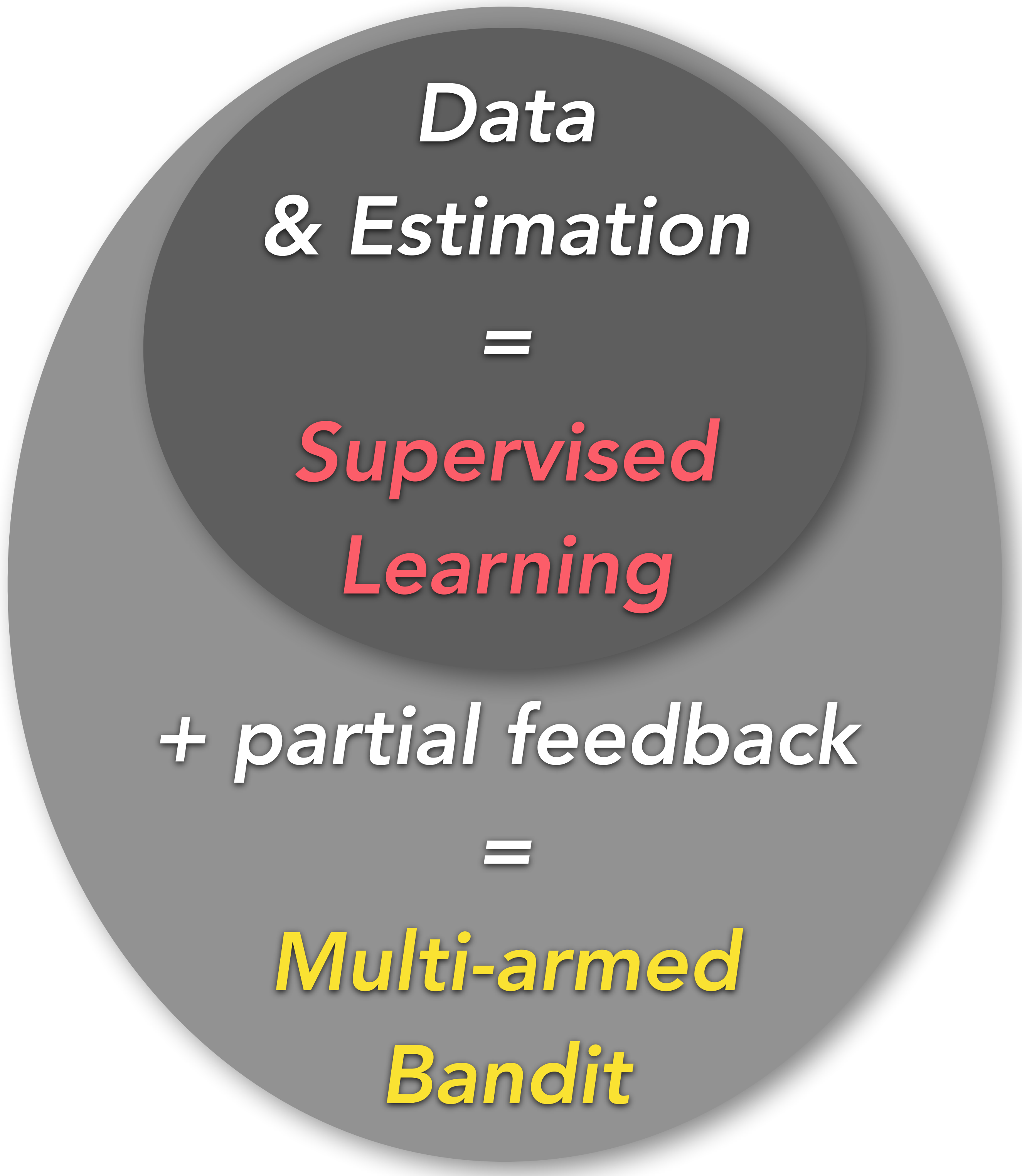
bit.ly/rpf_nips
@ianosband

Reinforcement Learning

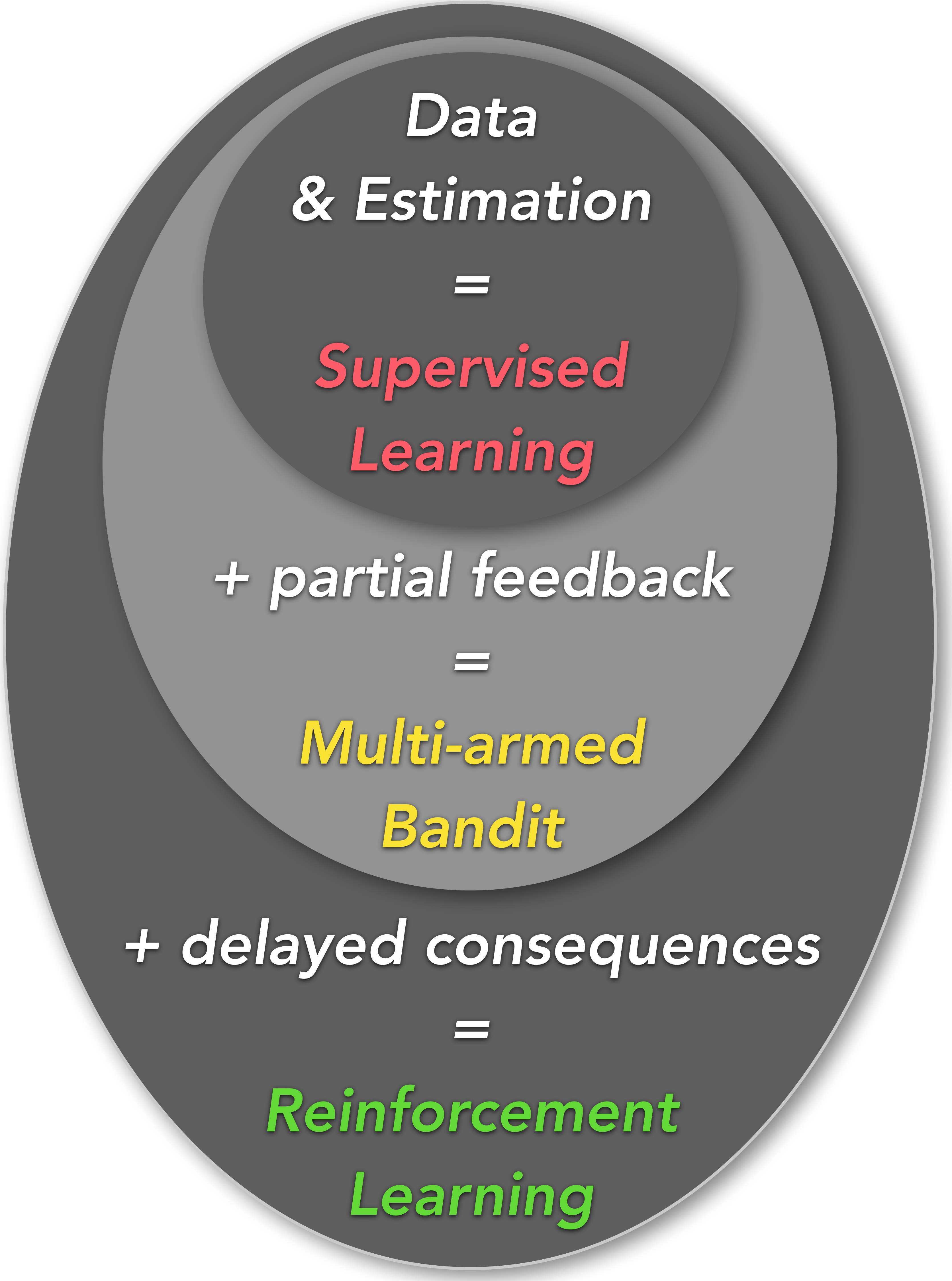
Data
& Estimation
=
Supervised Learning

bit.ly/rpf_nips
@ianosband

Reinforcement Learning

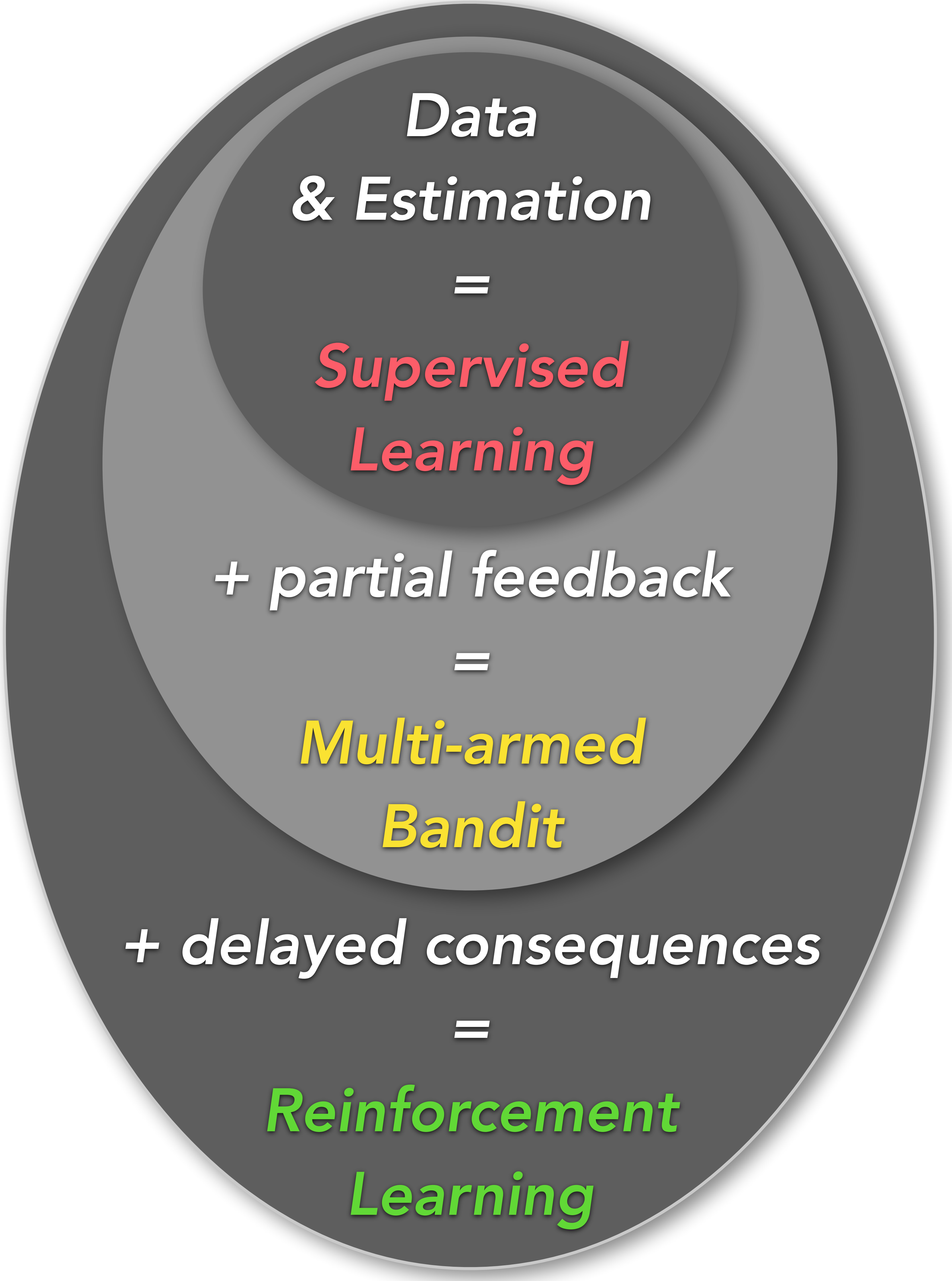


Reinforcement Learning



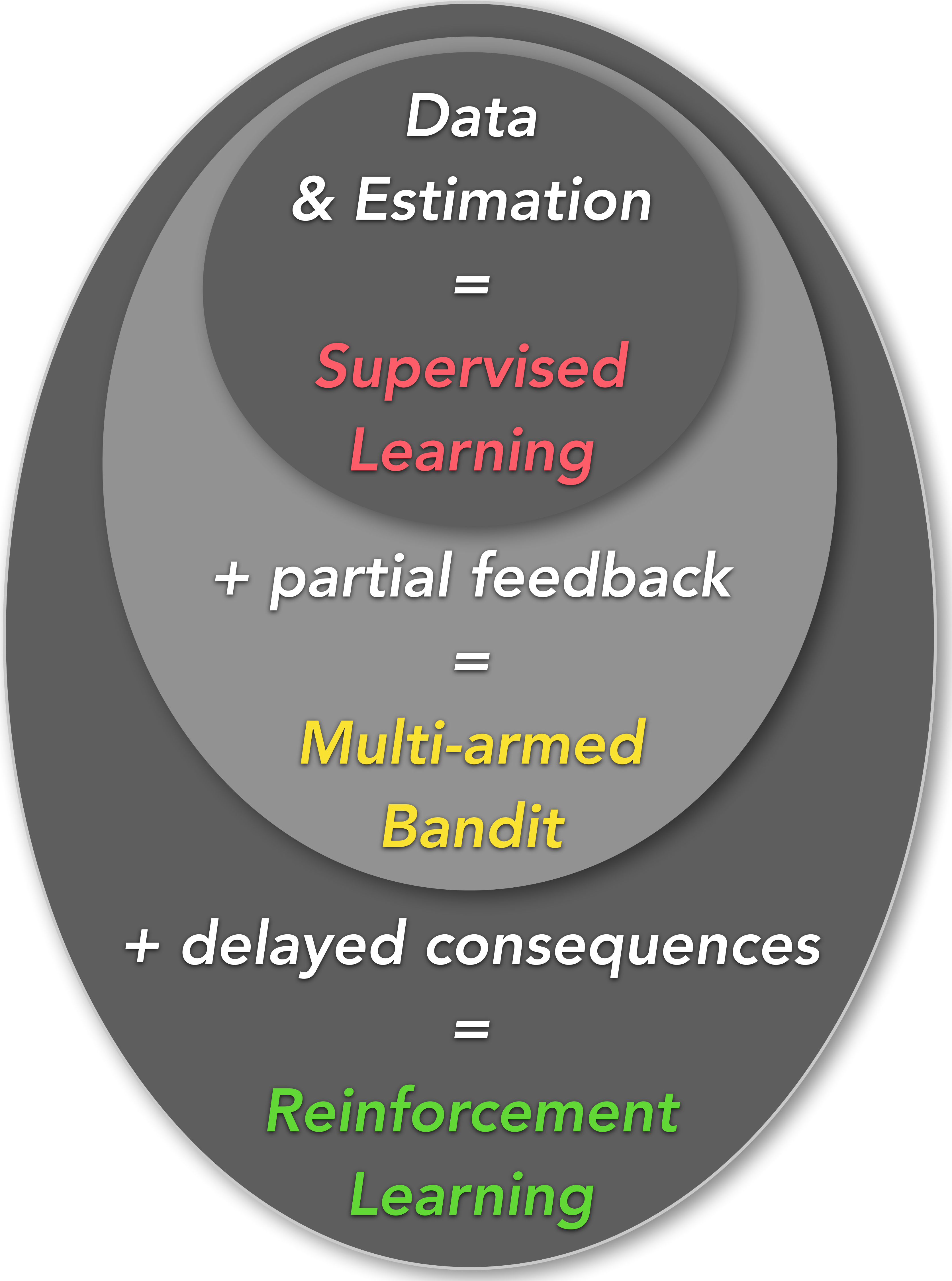
Reinforcement Learning

- *"Sequential decision making under uncertainty."*



Reinforcement Learning

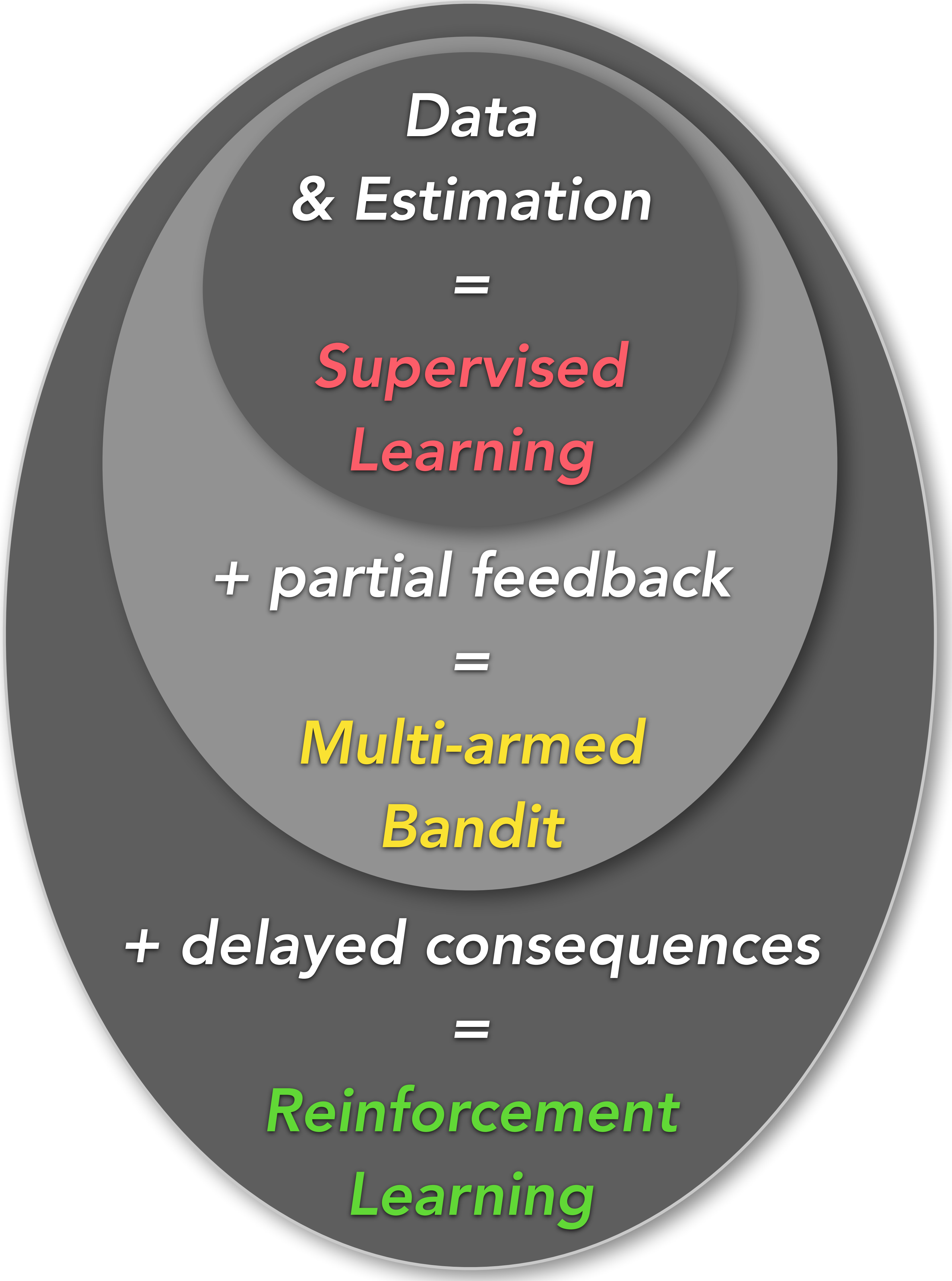
- *"Sequential decision making under uncertainty."*



Reinforcement Learning

- *"Sequential decision making under uncertainty."*

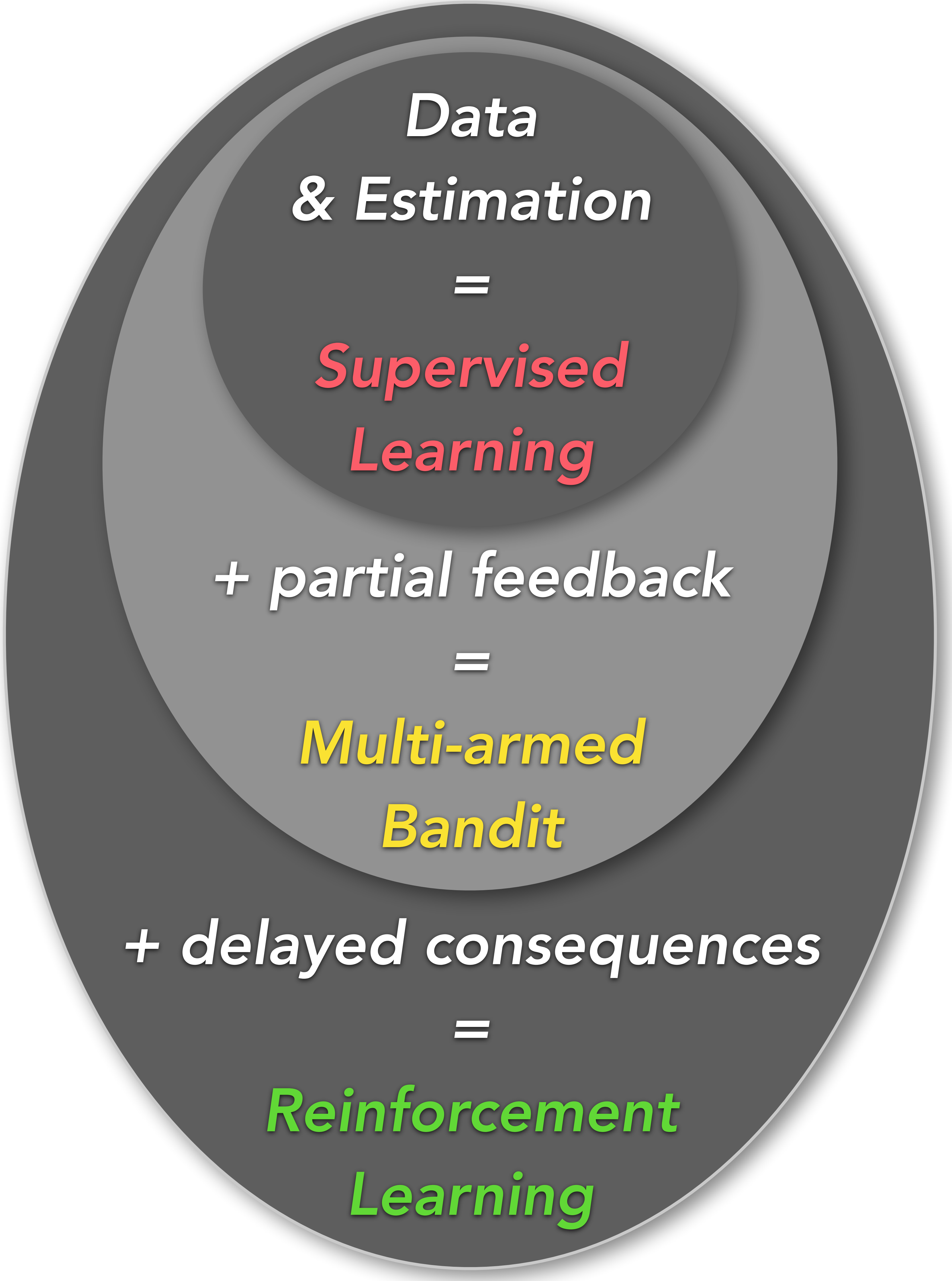
- Three necessary building blocks:



Reinforcement Learning

- *"Sequential decision making under uncertainty."*

- Three necessary building blocks:
    1. **Generalization**

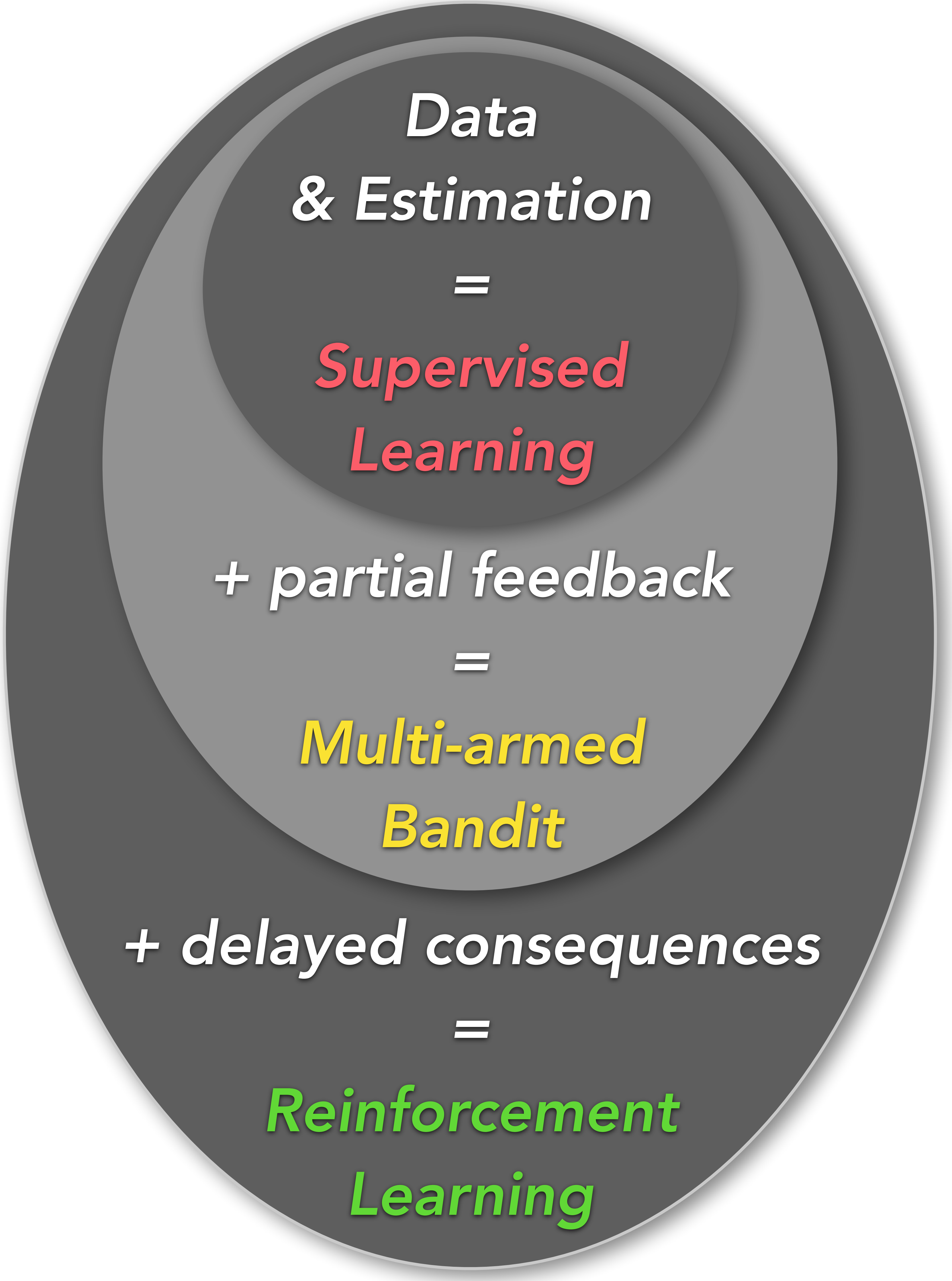


Reinforcement Learning

bit.ly/rpf_nips
@ianosband

- *"Sequential decision making under uncertainty."*

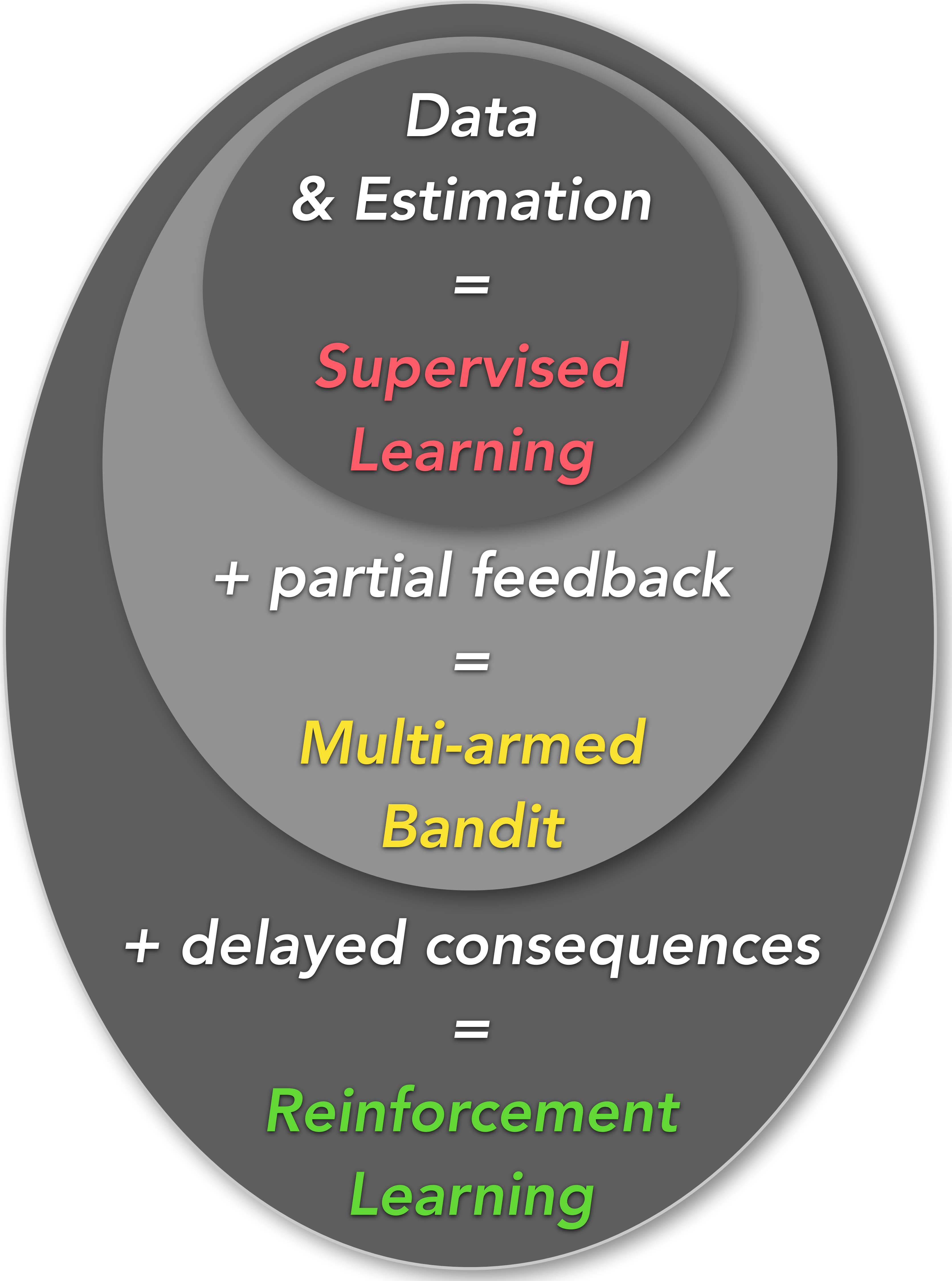
- Three necessary building blocks:
    1. **Generalization**
    2. **Exploration vs Exploitation**



Reinforcement Learning

- *"Sequential decision making under uncertainty."*

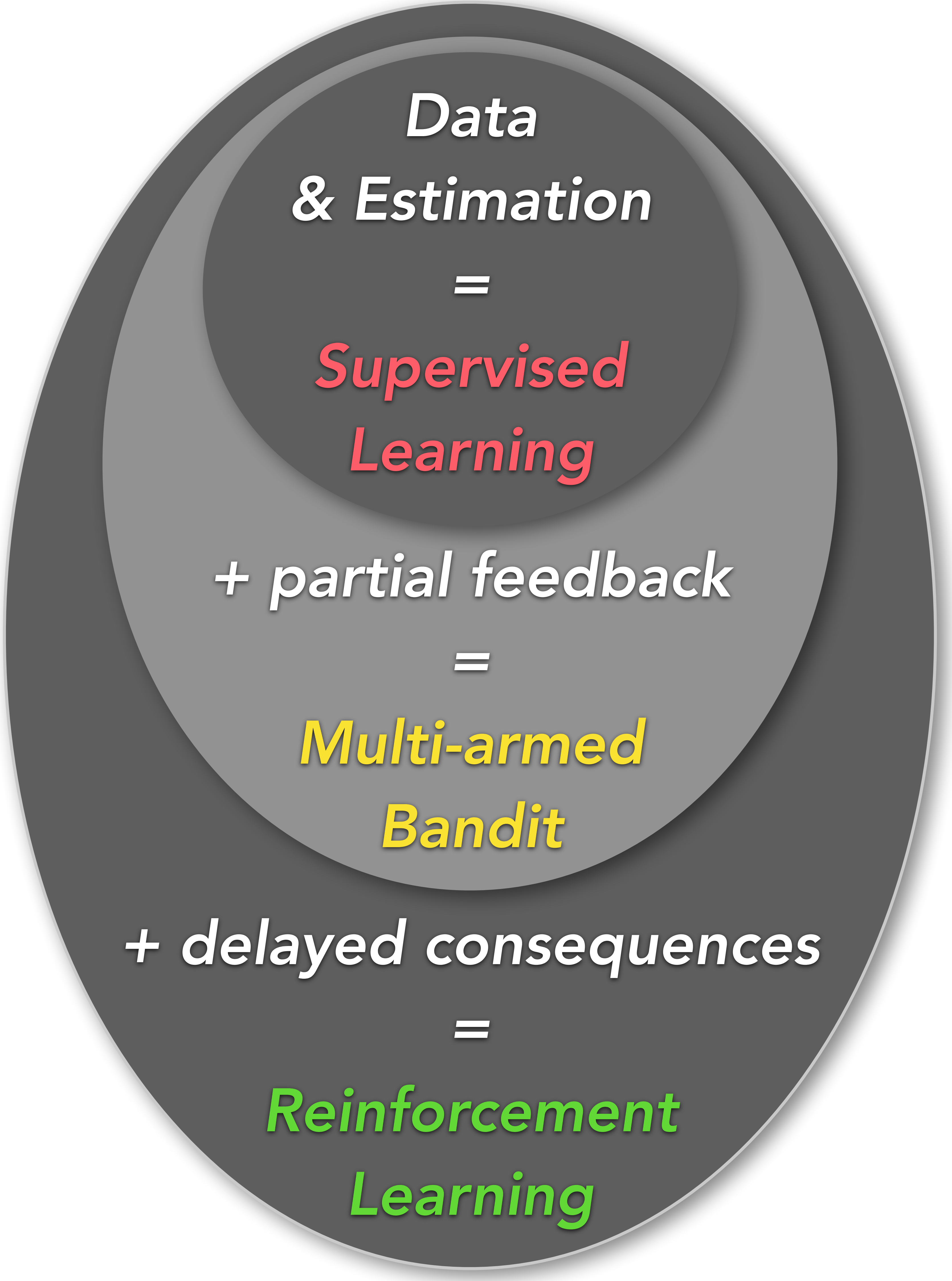
- Three necessary building blocks:
    1. **Generalization**
    2. **Exploration vs Exploitation**
    3. **Credit assignment**



Reinforcement Learning

- *"Sequential decision making under uncertainty."*

- Three necessary building blocks:
    1. **Generalization**
    2. **Exploration vs Exploitation**
    3. **Credit assignment**

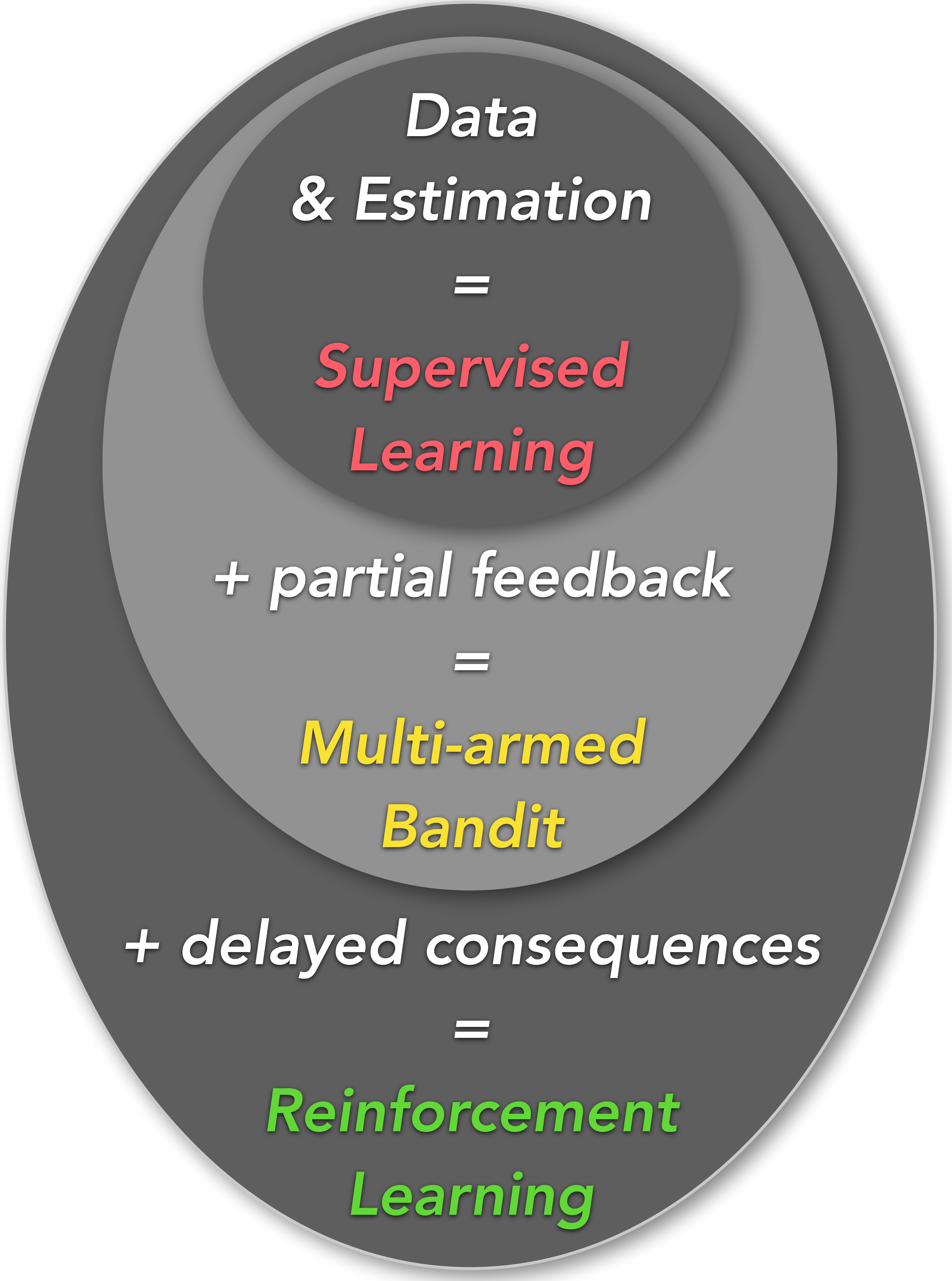


Reinforcement Learning

- *"Sequential decision making under uncertainty."*

- Three necessary building blocks:
  1. **Generalization**
  2. **Exploration vs Exploitation**
  3. **Credit assignment**

- As a field, we are pretty good at combining any 2 of these 3.
  … but we need practical solutions that combine them all.



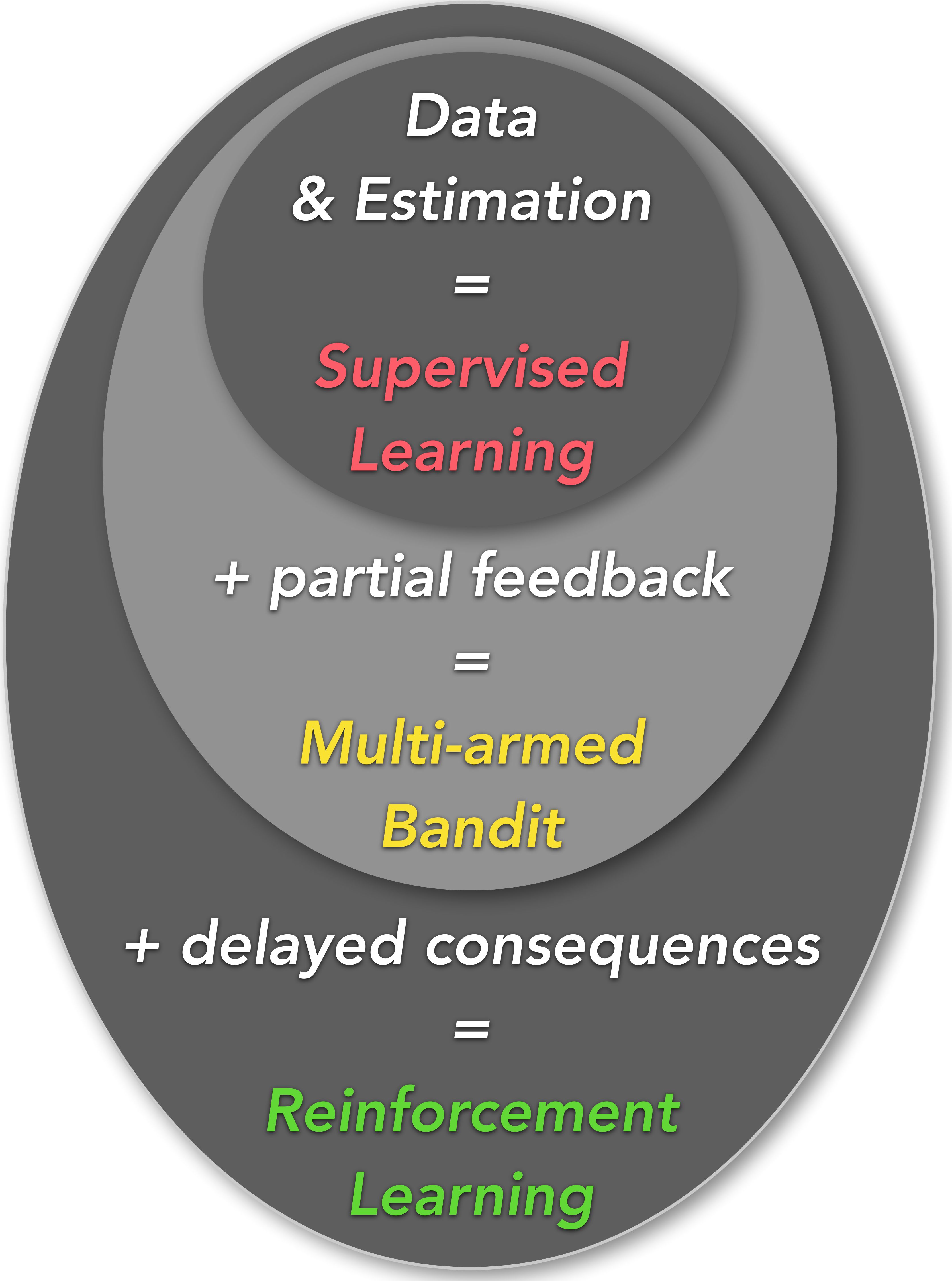
Reinforcement Learning

- *"Sequential decision making under uncertainty."*

- Three necessary building blocks:
  1. **Generalization**
  2. **Exploration vs Exploitation**
  3. **Credit assignment**

- As a field, we are pretty good at combining any 2 of these 3.
  … but we need practical solutions that combine them all.

We need effective uncertainty estimates for Deep RL



Estimating uncertainty in deep RL

Estimating uncertainty in deep RL

Dropout sampling

Estimating uncertainty in deep RL

Dropout sampling

"Dropout sample ≈ posterior sample" (Gal+Gharamani 2015)

Estimating uncertainty in deep RL

Dropout sampling

"Dropout sample ≈ posterior sample" (Gal+Gharamani 2015)

Dropout rate does not concentrate with the data.

Estimating uncertainty in deep RL

Dropout sampling

"Dropout sample ≈ posterior sample" (Gal+Gharamani 2015)

Dropout rate does not concentrate with the data.

Even "concrete" dropout not necessarily right rate.

Estimating uncertainty in deep RL

Dropout sampling

Variational inference

"Dropout sample ≈ posterior sample" (Gal+Gharamani 2015)

Dropout rate does not concentrate with the data.

Even "concrete" dropout not necessarily right rate.

Estimating uncertainty in deep RL

Dropout sampling

"Dropout sample ≈ posterior sample" (Gal+Gharamani 2015)

Dropout rate does not concentrate with the data.

Even "concrete" dropout not necessarily right rate.

Variational inference

Apply VI to Bellman error as if it was an i.i.d. supervised loss.

Estimating uncertainty in deep RL

Dropout sampling	**Variational inference**
"Dropout sample ≈ posterior sample" (Gal+Gharamani 2015)	*Apply VI to Bellman error as if it was an i.i.d. supervised loss.*
Dropout rate does not concentrate with the data.	Bellman error: $Q(s,a) = r + \gamma \max_{\alpha} Q(s', \alpha)$ Uncertainty in Q ➡ correlated TD loss. VI on i.i.d. model does not propagate uncertainty.
Even "concrete" dropout not necessarily right rate.	

Estimating uncertainty in deep RL

Dropout sampling	Variational inference	Distributional RL
"Dropout sample ≈ posterior sample" (Gal+Gharamani 2015)	*Apply VI to Bellman error as if it was an i.i.d. supervised loss.*	
Dropout rate does not concentrate with the data.	Bellman error: $Q(s,a) = r + \gamma \max_{\alpha} Q(s', \alpha)$ Uncertainty in Q ➡ correlated TD loss. VI on i.i.d. model does not propagate uncertainty.	
Even "concrete" dropout not necessarily right rate.		

Estimating uncertainty in deep RL

Dropout sampling	Variational inference	Distributional RL
"Dropout sample ≈ posterior sample" (Gal+Gharamani 2015)	*Apply VI to Bellman error as if it was an i.i.d. supervised loss.*	*Models Q-value as a distribution, rather than point estimate.*
Dropout rate does not concentrate with the data.	Bellman error: $Q(s,a) = r + \gamma \max_{\alpha} Q(s', \alpha)$ Uncertainty in Q ➡ correlated TD loss. VI on i.i.d. model does not propagate uncertainty.	
Even "concrete" dropout not necessarily right rate.		

Estimating uncertainty in deep RL

Dropout sampling	Variational inference	Distributional RL
"Dropout sample ≈ posterior sample" (Gal+Gharamani 2015)	*Apply VI to Bellman error as if it was an i.i.d. supervised loss.*	*Models Q-value as a distribution, rather than point estimate.*
Dropout rate does not concentrate with the data.	Bellman error: $Q(s,a) = r + \gamma \max_{\alpha} Q(s', \alpha)$	This distribution != posterior uncertainty.
Even "concrete" dropout not necessarily right rate.	Uncertainty in Q ➡ correlated TD loss.	
	VI on i.i.d. model does not propagate uncertainty.	

Estimating uncertainty in deep RL

Dropout sampling	Variational inference	Distributional RL
"Dropout sample ≈ posterior sample" (Gal+Gharamani 2015)	*Apply VI to Bellman error as if it was an i.i.d. supervised loss.*	*Models Q-value as a distribution, rather than point estimate.*
Dropout rate does not concentrate with the data.	Bellman error: $Q(s,a) = r + \gamma \max_{\alpha} Q(s',\alpha)$ Uncertainty in Q ➡ correlated TD loss. VI on i.i.d. model does not propagate uncertainty.	This distribution != posterior uncertainty.
Even "concrete" dropout not necessarily right rate.		*Aleatoric* vs *Epistemic* … it's not the right thing for exploration.

Estimating uncertainty in deep RL

Dropout sampling

"Dropout sample ≈ posterior sample" (Gal+Gharamani 2015)

Dropout rate does not concentrate with the data.

Even "concrete" dropout not necessarily right rate.

Variational inference

Apply VI to Bellman error as if it was an i.i.d. supervised loss.

Bellman error:

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

Uncertainty in Q ➡ correlated TD loss.

VI on i.i.d. model does not propagate uncertainty.

Distributional RL

Models Q-value as a distribution, rather than point estimate.

This distribution != posterior uncertainty.

Aleatoric vs *Epistemic* … it's *not* the right thing for exploration.

Count-based density

Estimating uncertainty in deep RL

Dropout sampling	**Variational inference**	**Distributional RL**	**Count-based density**
"Dropout sample ≈ posterior sample" (Gal+Gharamani 2015)	*Apply VI to Bellman error as if it was an i.i.d. supervised loss.*	*Models Q-value as a distribution, rather than point estimate.*	*Estimate number of "visit counts" to state, add bonus.*
Dropout rate does not concentrate with the data.	Bellman error: $Q(s,a) = r + \gamma \max_{\alpha} Q(s', \alpha)$ Uncertainty in Q ➡ correlated TD loss. VI on i.i.d. model does not propagate uncertainty.	This distribution != posterior uncertainty.	
Even "concrete" dropout not necessarily right rate.		*Aleatoric* vs *Epistemic* … it's not the right thing for exploration.	

bit.ly/rpf_nips
@ianosband

Estimating uncertainty in deep RL

Dropout sampling	Variational inference	Distributional RL	Count-based density
"Dropout sample ≈ posterior sample" (Gal+Gharamani 2015)	*Apply VI to Bellman error as if it was an i.i.d. supervised loss.*	*Models Q-value as a distribution, rather than point estimate.*	*Estimate number of "visit counts" to state, add bonus.*
Dropout rate does not concentrate with the data.	Bellman error: $Q(s,a) = r + \gamma \max_{\alpha} Q(s', \alpha)$ Uncertainty in Q ➡ correlated TD loss. VI on i.i.d. model does not propagate uncertainty.	This distribution != posterior uncertainty.	The "density model" has nothing to do with the actual task.
Even "concrete" dropout not necessarily right rate.		*Aleatoric* vs *Epistemic* … it's not the right thing for exploration.	

Estimating uncertainty in deep RL

Dropout sampling	**Variational inference**	**Distributional RL**	**Count-based density**
"Dropout sample ≈ posterior sample" (Gal+Gharamani 2015)	*Apply VI to Bellman error as if it was an i.i.d. supervised loss.*	*Models Q-value as a distribution, rather than point estimate.*	*Estimate number of "visit counts" to state, add bonus.*
Dropout rate does not concentrate with the data.	Bellman error: $Q(s,a) = r + \gamma \max_{\alpha} Q(s', \alpha)$ Uncertainty in Q ➡ correlated TD loss. VI on i.i.d. model does not propagate uncertainty.	This distribution != posterior uncertainty.	The "density model" has nothing to do with the actual task.
Even "concrete" dropout not necessarily right rate.		*Aleatoric* vs *Epistemic* … it's not the right thing for exploration.	With generalization, state "visit count" != uncertainty.

Estimating uncertainty in deep RL

Dropout sampling	**Variational inference**	**Distributional RL**	**Count-based density**	**Bootstrap ensemble**
"Dropout sample ≈ posterior sample" (Gal+Gharamani 2015)	*Apply VI to Bellman error as if it was an i.i.d. supervised loss.*	*Models Q-value as a distribution, rather than point estimate.*	*Estimate number of "visit counts" to state, add bonus.*	
Dropout rate does not concentrate with the data.	Bellman error: $Q(s,a) = r + \gamma \max_{\alpha} Q(s', \alpha)$ Uncertainty in Q ➡ correlated TD loss. VI on i.i.d. model does not propagate uncertainty.	This distribution != posterior uncertainty.	The "density model" has nothing to do with the actual task.	
Even "concrete" dropout not necessarily right rate.		*Aleatoric* vs *Epistemic* … it's not the right thing for exploration.	With generalization, state "visit count" != uncertainty.	

Estimating uncertainty in deep RL

Dropout sampling	Variational inference	Distributional RL	Count-based density	Bootstrap ensemble
"Dropout sample ≈ posterior sample" (Gal+Gharamani 2015)	*Apply VI to Bellman error as if it was an i.i.d. supervised loss.*	*Models Q-value as a distribution, rather than point estimate.*	*Estimate number of "visit counts" to state, add bonus.*	*Train ensemble on noisy data - classic statistical procedure!*
Dropout rate does not concentrate with the data.	Bellman error: $Q(s,a) = r + \gamma \max_{\alpha} Q(s',\alpha)$ Uncertainty in Q ➡ correlated TD loss. VI on i.i.d. model does not propagate uncertainty.	This distribution != posterior uncertainty.	The "density model" has nothing to do with the actual task.	
Even "concrete" dropout not necessarily right rate.		*Aleatoric* vs *Epistemic* … it's not the right thing for exploration.	With generalization, state "visit count" != uncertainty.	

bit.ly/rpf_nips
@ianosband

Estimating uncertainty in deep RL

Dropout sampling

"Dropout sample ≈ posterior sample" (Gal+Gharamani 2015)

Dropout rate does not concentrate with the data.

Even "concrete" dropout not necessarily right rate.

Variational inference

Apply VI to Bellman error as if it was an i.i.d. supervised loss.

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

Uncertainty in Q ➡ correlated TD loss.

VI on i.i.d. model does not propagate uncertainty.

Distributional RL

Models Q-value as a distribution, rather than point estimate.

This distribution != posterior uncertainty.

Aleatoric vs *Epistemic* … it's not the right thing for exploration.

Count-based density

Estimate number of "visit counts" to state, add bonus.

The "density model" has nothing to do with the actual task.

With generalization, state "visit count" != uncertainty.

Bootstrap ensemble

Train ensemble on noisy data - classic statistical procedure!

No explicit "prior" mechanism for "intrinsic motivation"

bit.ly/rpf_nips
@ianosband

Estimating uncertainty in deep RL

Dropout sampling	**Variational inference**	**Distributional RL**	**Count-based density**	**Bootstrap ensemble**
"Dropout sample ≈ posterior sample" (Gal+Gharamani 2015)	*Apply VI to Bellman error as if it was an i.i.d. supervised loss.*	*Models Q-value as a distribution, rather than point estimate.*	*Estimate number of "visit counts" to state, add bonus.*	*Train ensemble on noisy data - classic statistical procedure!*
Dropout rate does not concentrate with the data.	Bellman error: $Q(s,a) = r + \gamma \max_{\alpha} Q(s', \alpha)$ Uncertainty in Q ➡ correlated TD loss. VI on i.i.d. model does not propagate uncertainty.	This distribution != posterior uncertainty.	The "density model" has nothing to do with the actual task.	No explicit "prior" mechanism for "intrinsic motivation"
Even "concrete" dropout not necessarily right rate.		*Aleatoric* vs *Epistemic* … it's not the right thing for exploration.	With generalization, state "visit count" != uncertainty.	If you've never seen a reward, why would the agent explore?

Randomized prior functions

Randomized prior functions

- **Key idea:** add a random untrainable “prior” function to each member of the ensemble.

Randomized prior functions

bit.ly/rpf_nips
@ianosband

- **Key idea:** add a random untrainable “prior” function to each member of the ensemble.

$$Q_\theta(x) = \underbrace{f_\theta(x)}_{\text{trainable}} + \underbrace{p(x)}_{\text{prior}}.$$

Randomized prior functions

bit.ly/rpf_nips
@ianosband

- **Key idea:** add a random untrainable "prior" function to each member of the ensemble.

$$Q_\theta(x) = \underbrace{f_\theta(x)}_{\text{trainable}} + \underbrace{p(x)}_{\text{prior}}.$$

- Visualize effects in 1D regression:

Randomized prior functions

- **Key idea:** add a random untrainable "prior" function to each member of the ensemble.

$$Q_\theta(x) = \underbrace{f_\theta(x)}_{\text{trainable}} + \underbrace{p(x)}_{\text{prior}}.$$

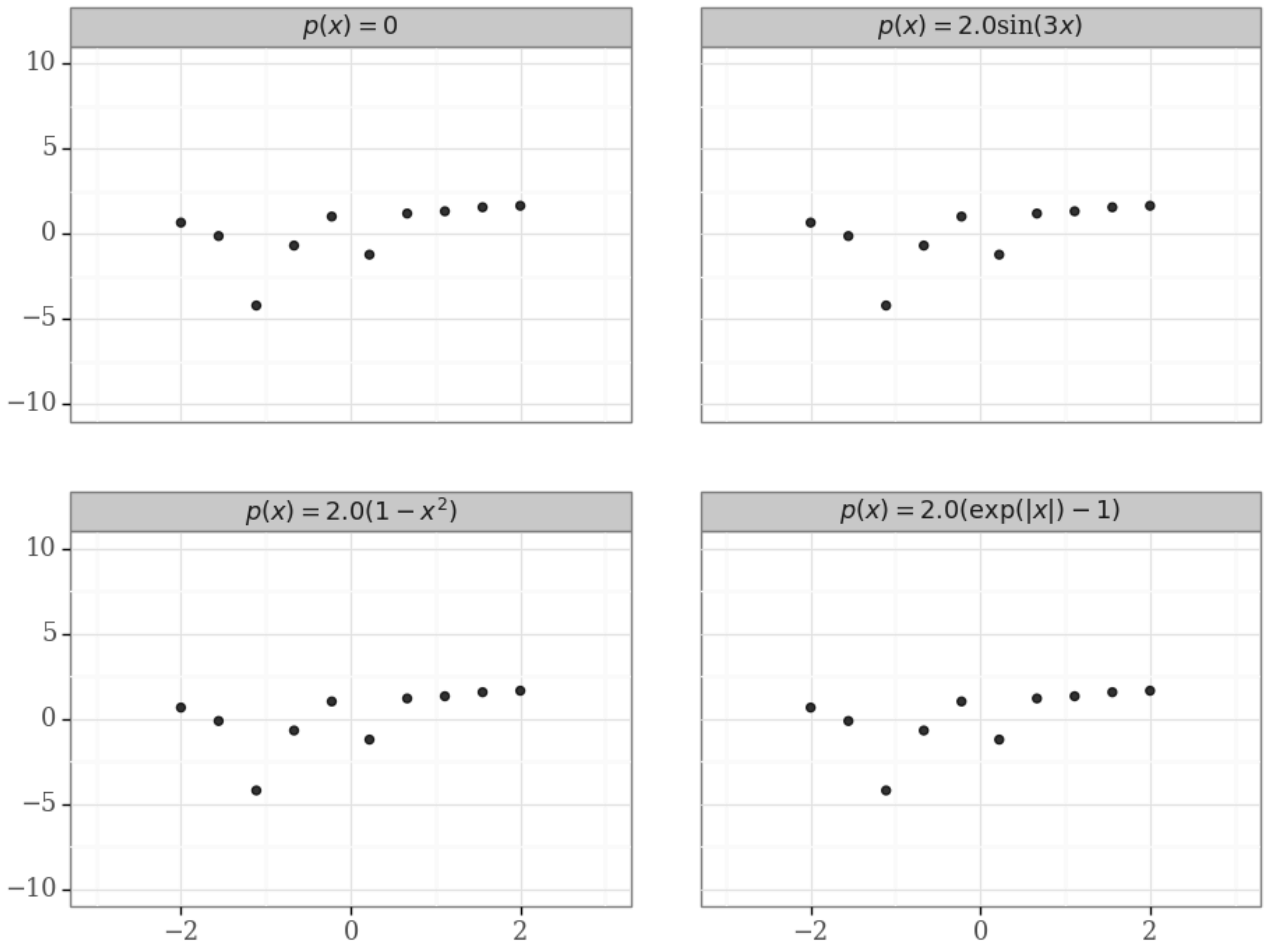
- Visualize effects in 1D regression:
  - Training data **(x,y) black points.**

Randomized prior functions

- **Key idea:** add a random untrainable "prior" function to each member of the ensemble.

$$Q_\theta(x) = \underbrace{f_\theta(x)}_{\text{trainable}} + \underbrace{p(x)}_{\text{prior}}.$$

- Visualize effects in 1D regression:
  - Training data **(x,y) black points.**



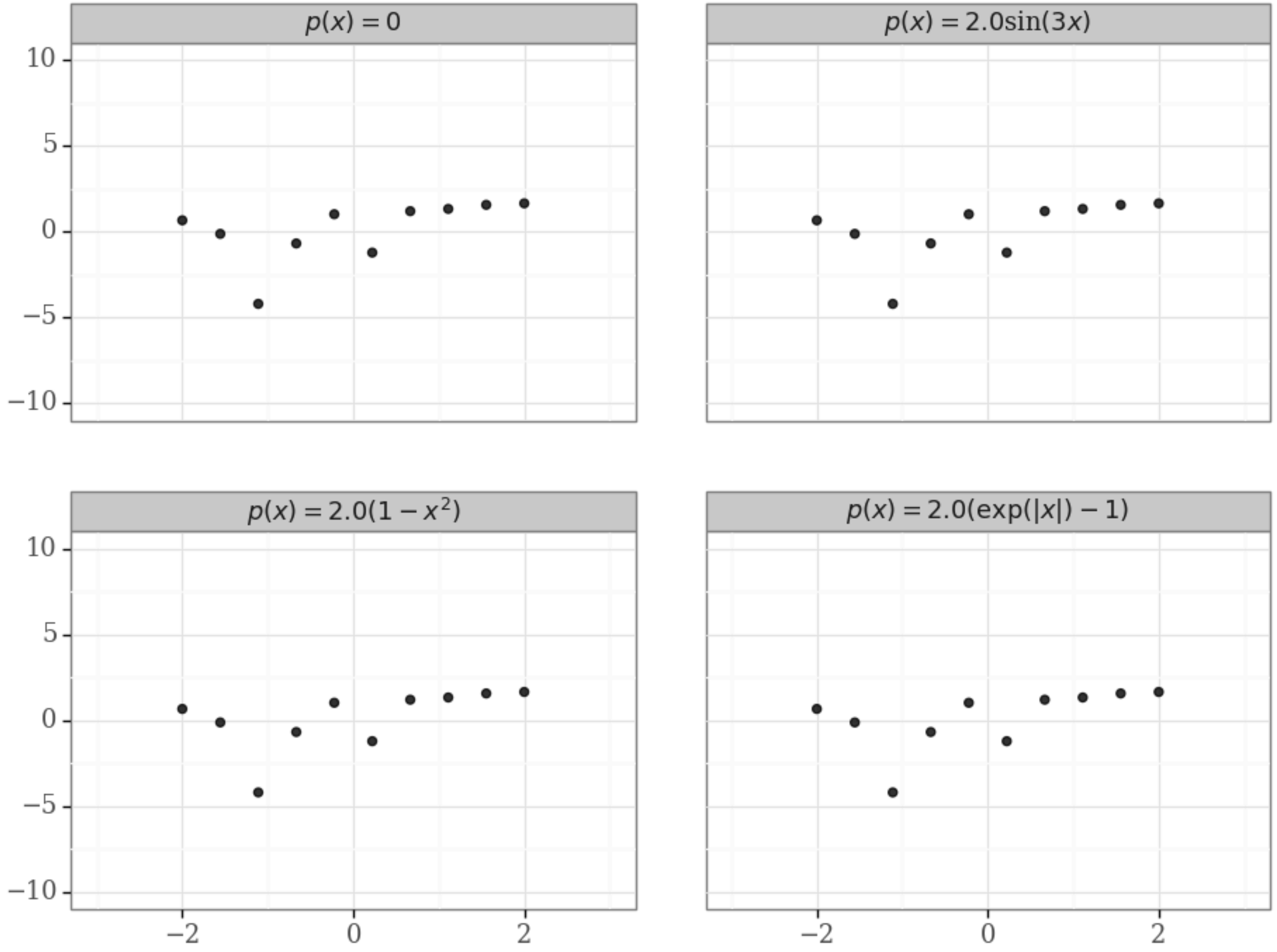
Randomized prior functions

bit.ly/rpf_nips
@ianosband

- **Key idea:** add a random untrainable "prior" function to each member of the ensemble.

$$Q_\theta(x) = \underbrace{f_\theta(x)}_{\text{trainable}} + \underbrace{p(x)}_{\text{prior}}.$$

- Visualize effects in 1D regression:
  - Training data **(x,y) black points.**
  - Prior function **p(x) blue line.**



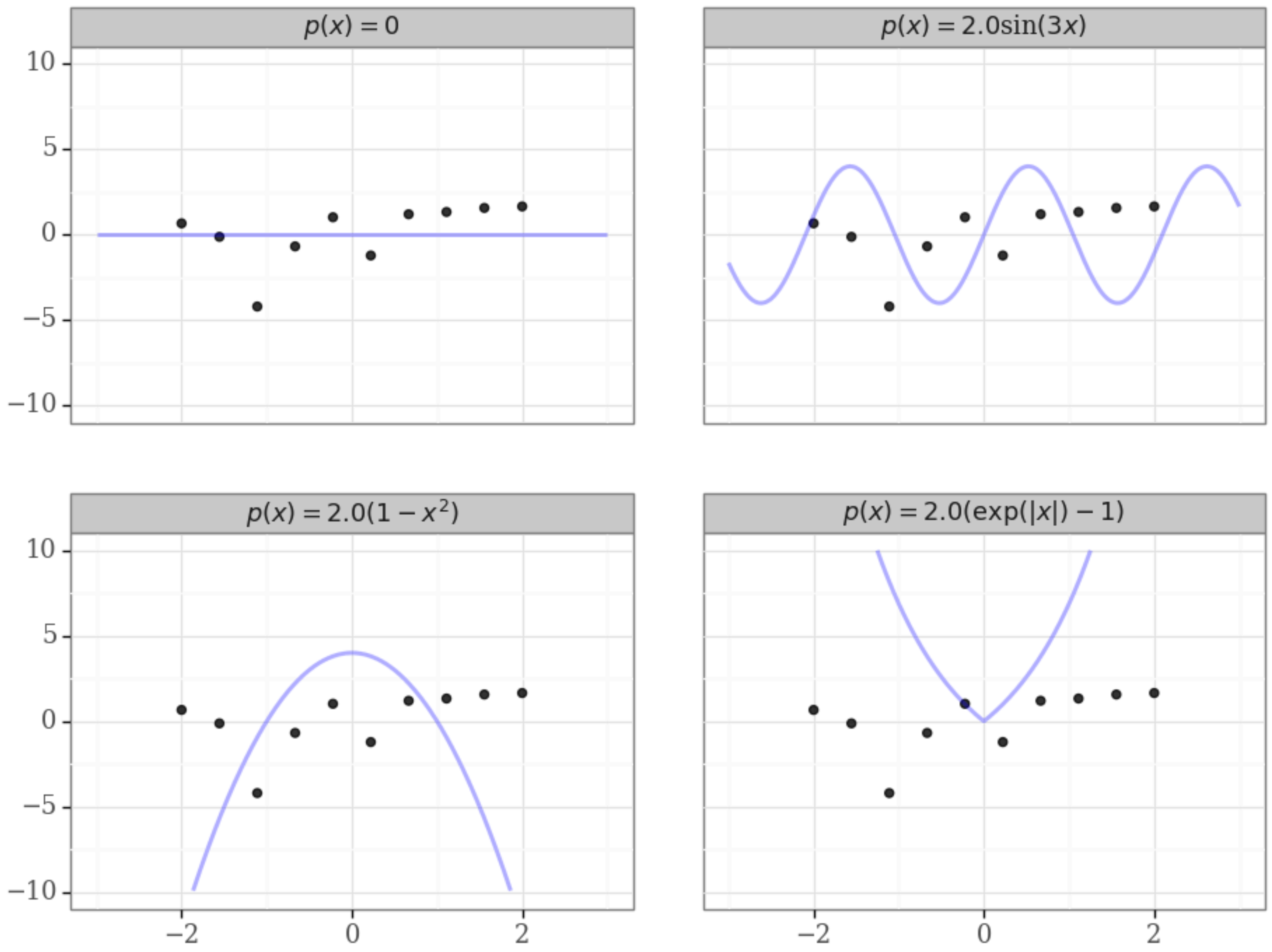
Randomized prior functions

bit.ly/rpf_nips
@ianosband

- **Key idea:** add a random untrainable "prior" function to each member of the ensemble.

$$Q_\theta(x) = \underbrace{f_\theta(x)}_{\text{trainable}} + \underbrace{p(x)}_{\text{prior}}.$$

- Visualize effects in 1D regression:
  - Training data **(x,y) black points.**
  - Prior function **p(x) blue line.**



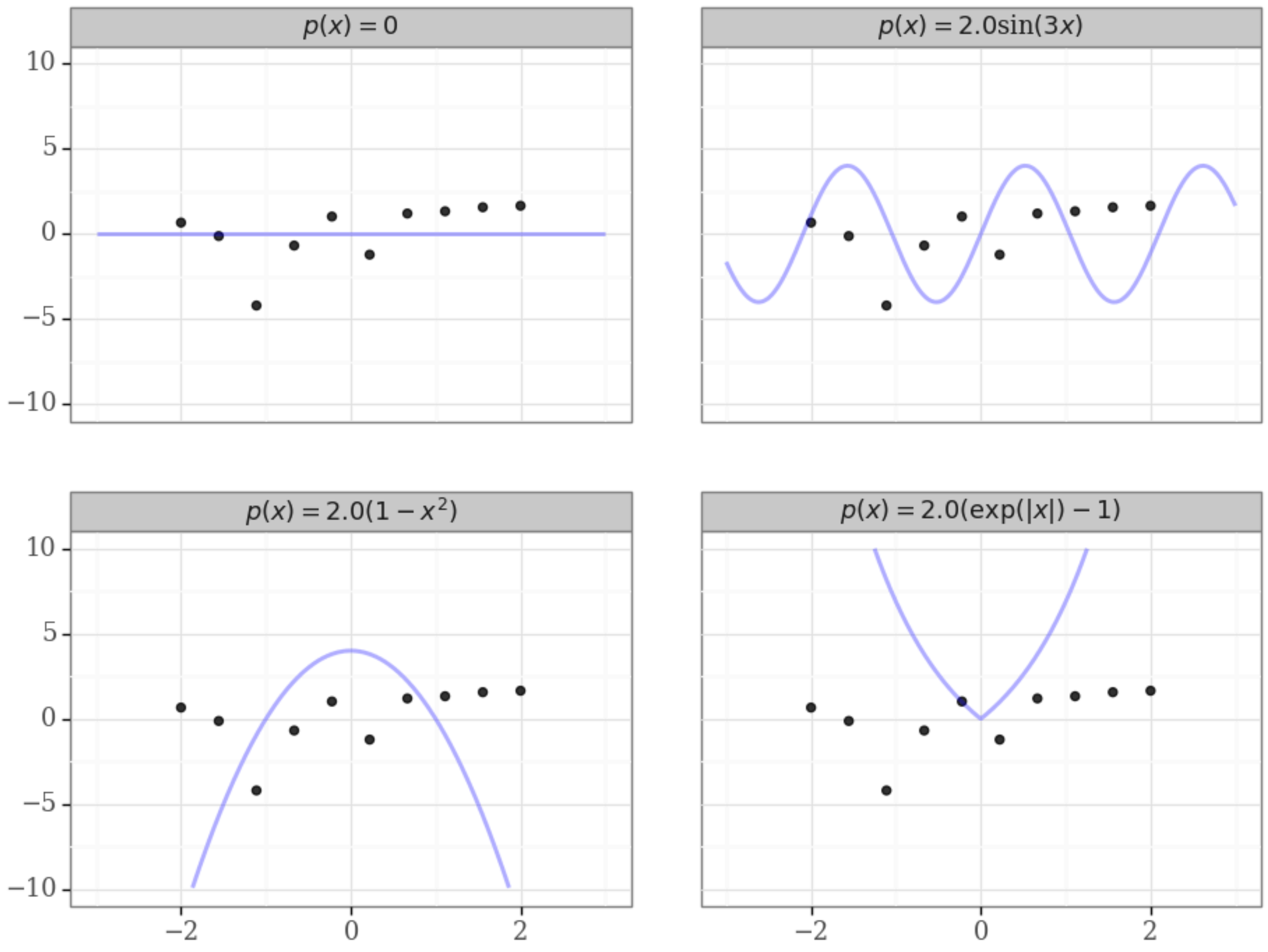
Randomized prior functions

bit.ly/rpf_nips
@ianosband

- **Key idea:** add a random untrainable "prior" function to each member of the ensemble.

$$Q_\theta(x) = \underbrace{f_\theta(x)}_{\text{trainable}} + \underbrace{p(x)}_{\text{prior}}.$$

- Visualize effects in 1D regression:
  - Training data **(x,y) black points.**
  - Prior function **p(x) blue line.**
  - Trainable function f(x) dotted line.



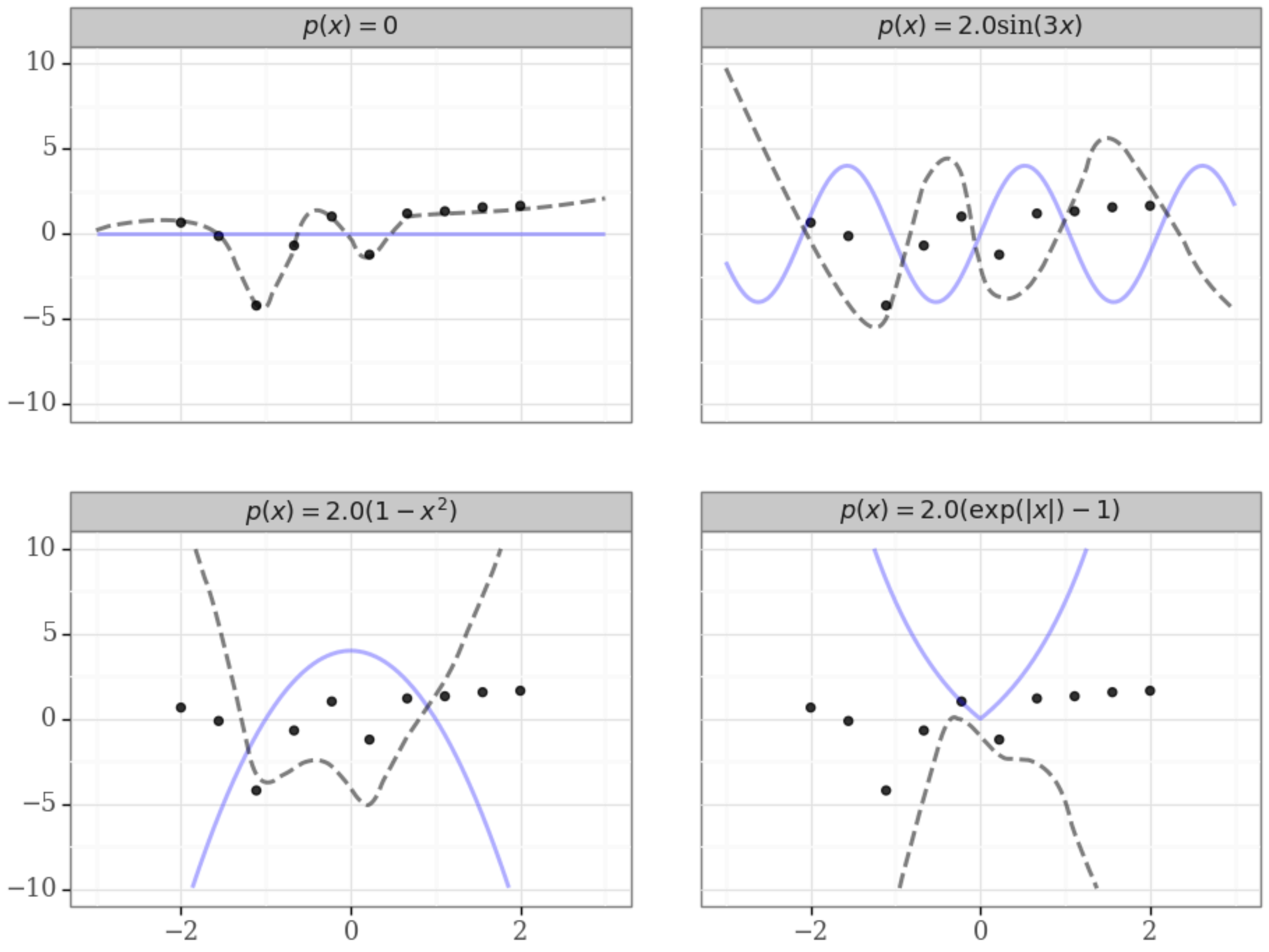
Randomized prior functions

bit.ly/rpf_nips
@ianosband

- **Key idea:** add a random untrainable "prior" function to each member of the ensemble.

$$Q_\theta(x) = \underbrace{f_\theta(x)}_{\text{trainable}} + \underbrace{p(x)}_{\text{prior}}.$$

- Visualize effects in 1D regression:
  - Training data **(x,y) black points.**
  - Prior function **p(x) blue line.**
  - Trainable function f(x) dotted line.



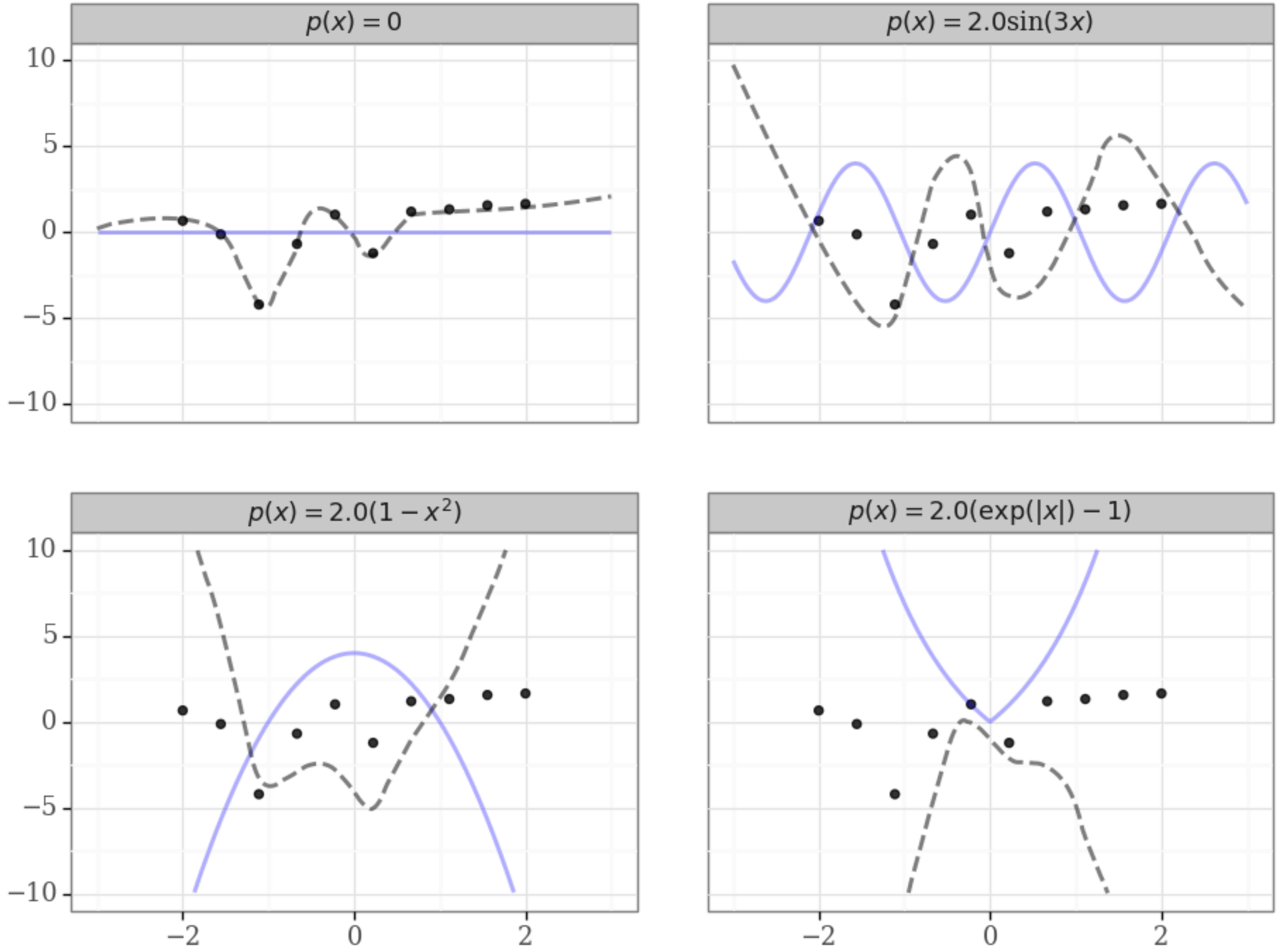
Randomized prior functions

bit.ly/rpf_nips
@ianosband

- **Key idea:** add a random untrainable "prior" function to each member of the ensemble.

$$Q_\theta(x) = \underbrace{f_\theta(x)}_{\text{trainable}} + \underbrace{p(x)}_{\text{prior}}.$$

- Visualize effects in 1D regression:
  - Training data **(x,y) black points.**
  - Prior function **p(x) blue line.**
  - Trainable function f(x) dotted line.
  - Output prediction **Q(x) red line.**



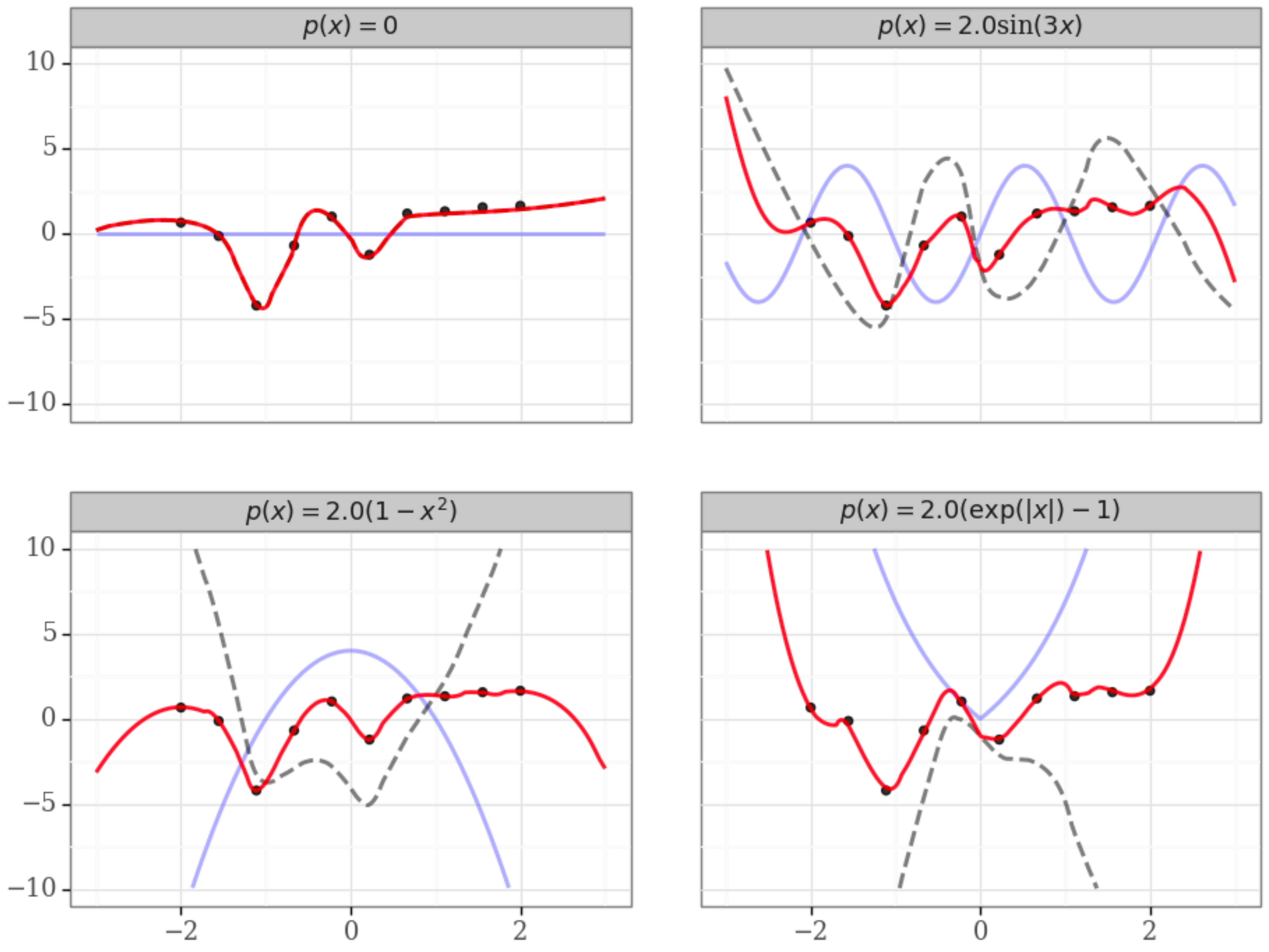
Randomized prior functions

bit.ly/rpf_nips
@ianosband

- **Key idea:** add a random untrainable "prior" function to each member of the ensemble.

$$Q_\theta(x) = \underbrace{f_\theta(x)}_{\text{trainable}} + \underbrace{p(x)}_{\text{prior}}.$$

- Visualize effects in 1D regression:
  - Training data **(x,y) black points.**
  - Prior function **p(x) blue line.**
  - Trainable function f(x) dotted line.
  - Output prediction **Q(x) red line.**



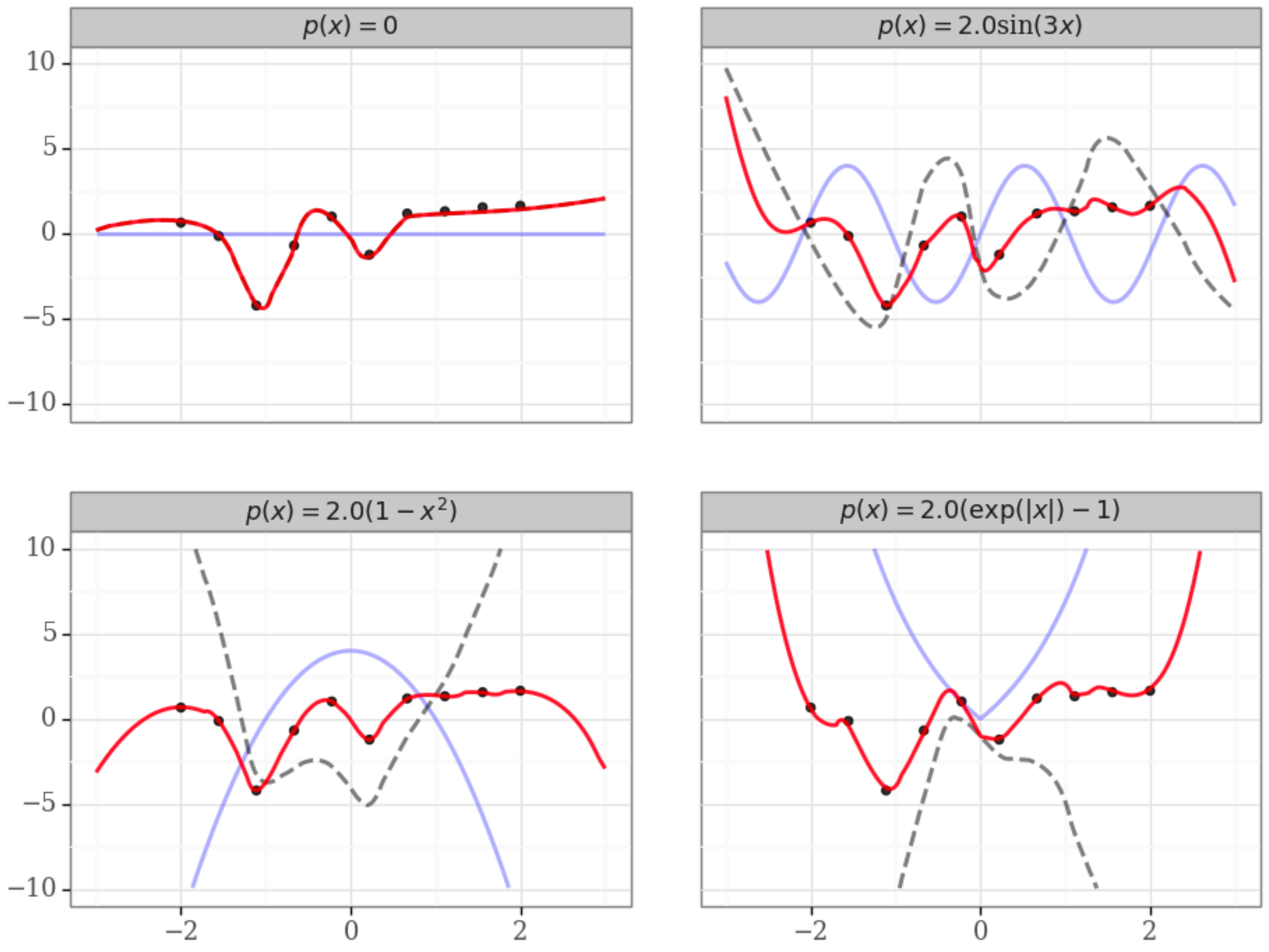
Randomized prior functions

- **Key idea:** add a random untrainable "prior" function to each member of the ensemble.

$$Q_\theta(x) = \underbrace{f_\theta(x)}_{\text{trainable}} + \underbrace{p(x)}_{\text{prior}}.$$

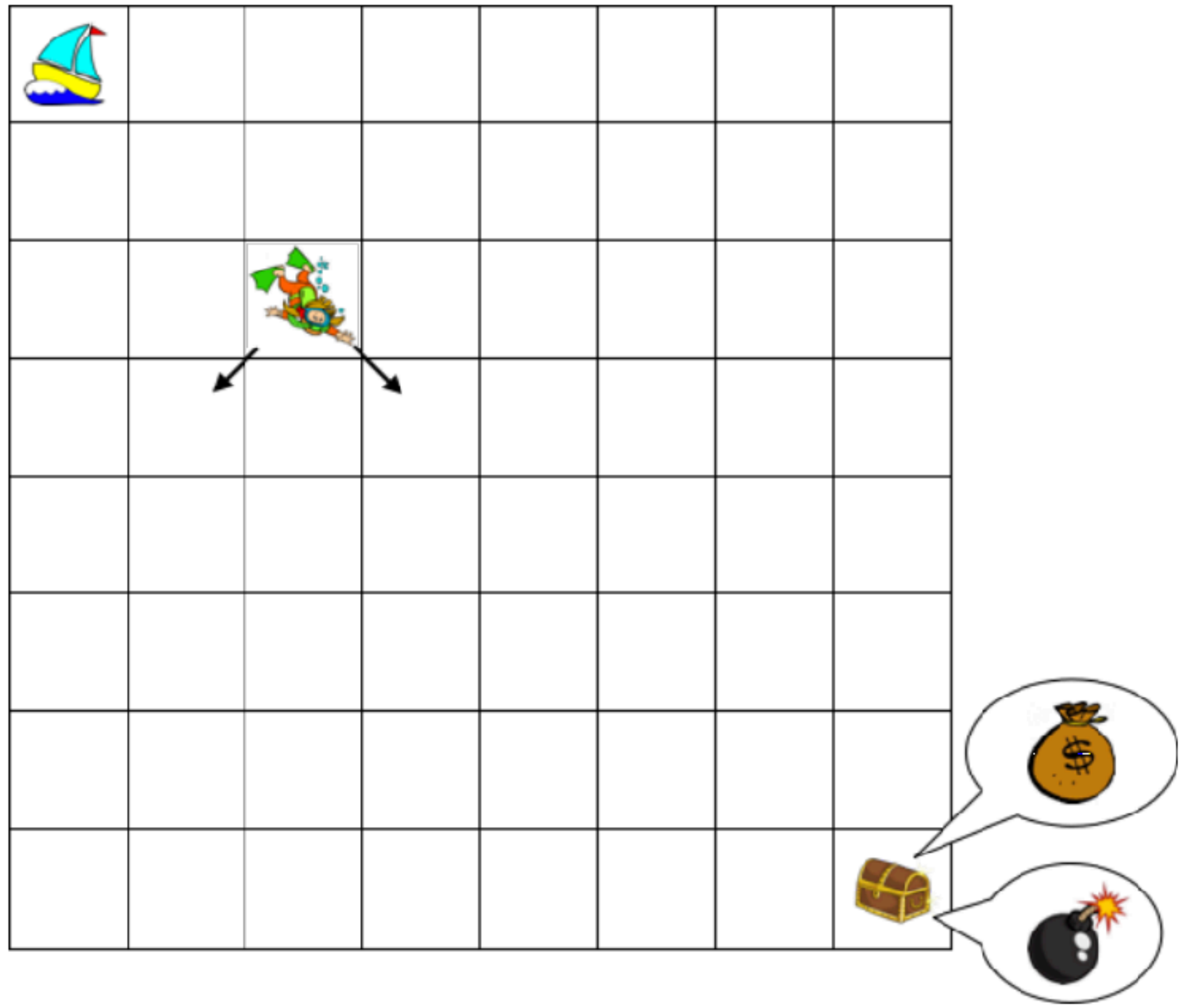
- Visualize effects in 1D regression:
  - Training data **(x,y) black points.**
  - Prior function **p(x) blue line.**
  - Trainable function f(x) dotted line.
  - Output prediction **Q(x) red line.**

Exact Bayes posterior for linear functions!



“Deep Sea” Exploration

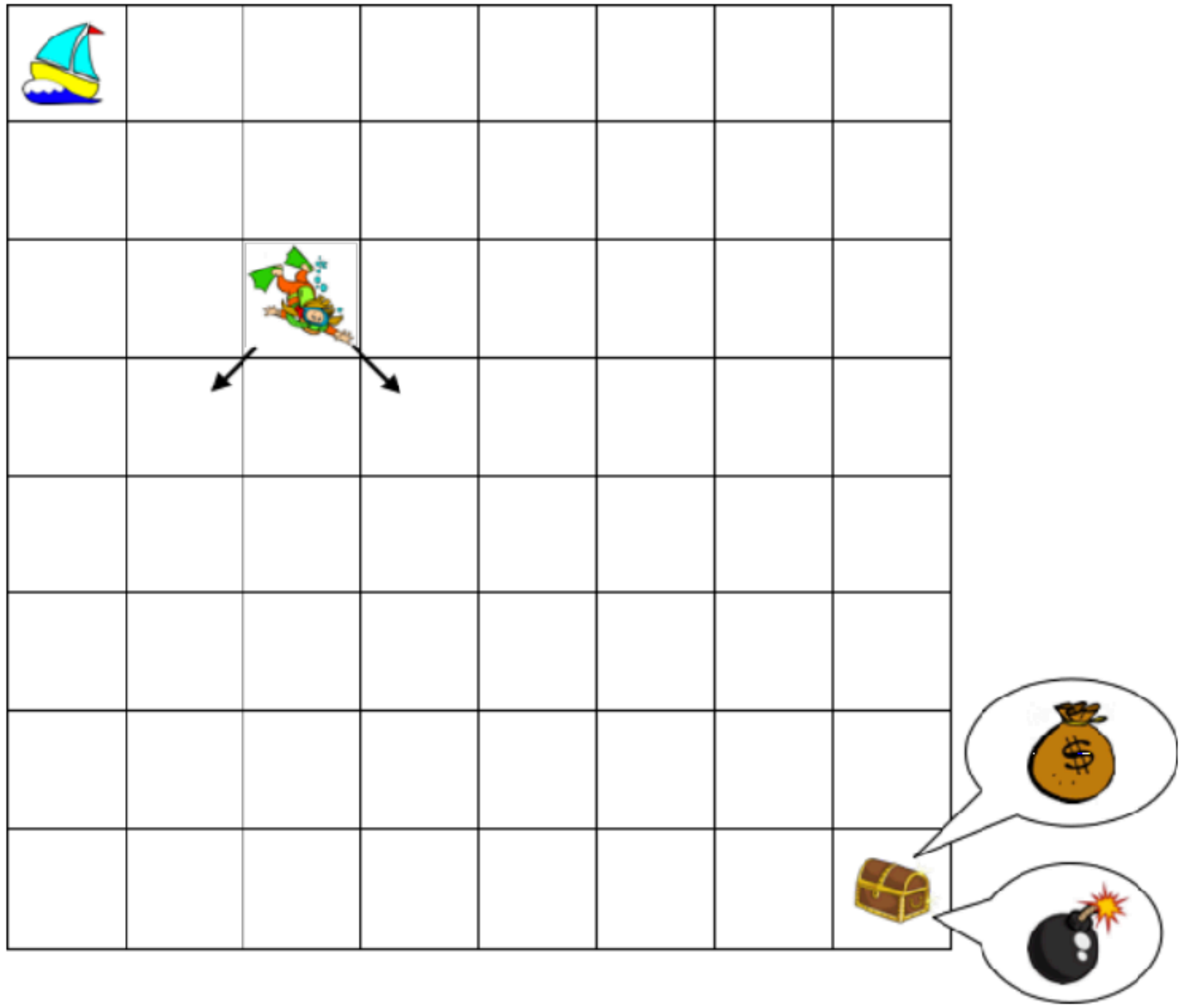
"Deep Sea" Exploration



"Deep Sea" Exploration

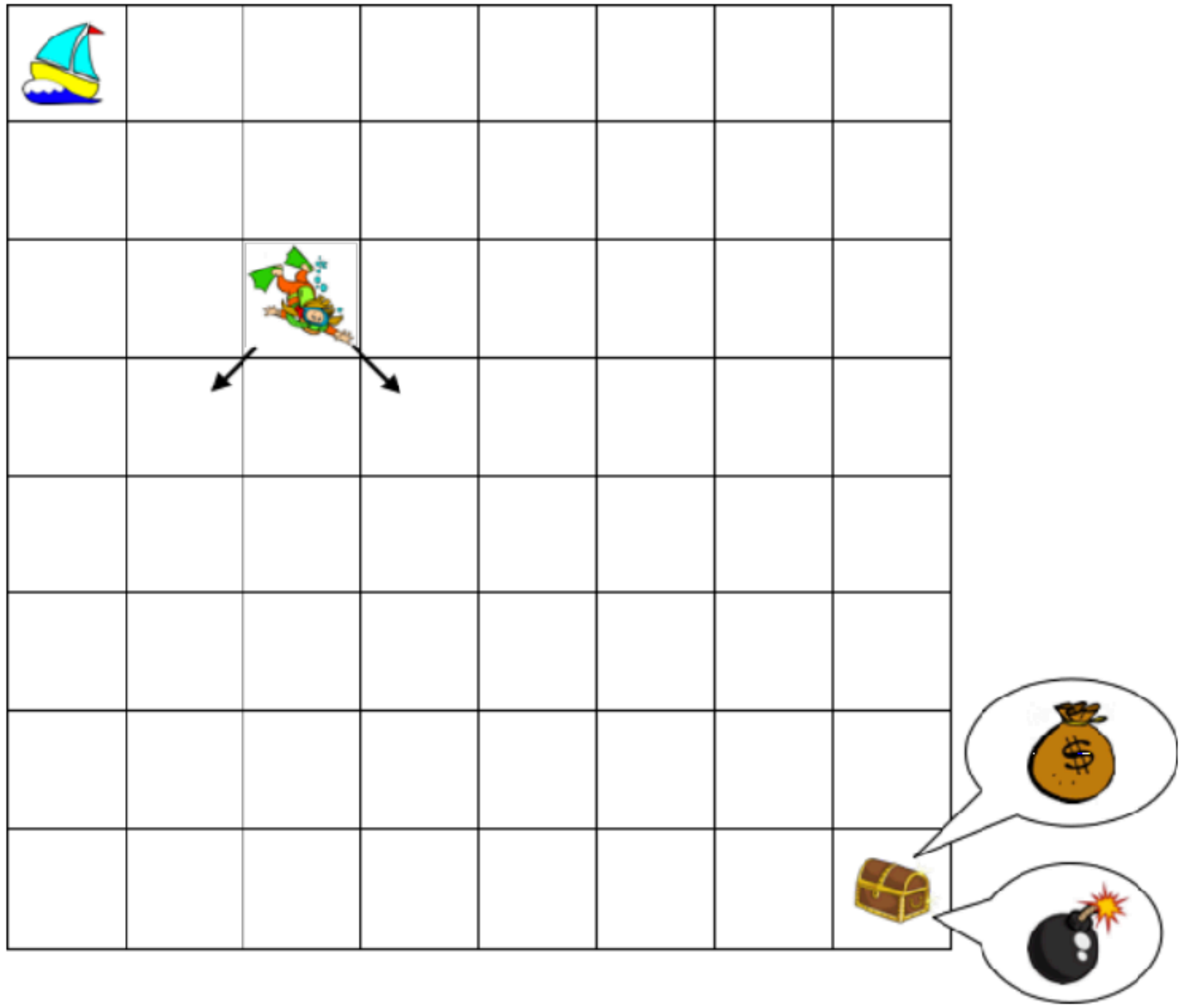
bit.ly/rpf_nips
@ianosband

- **Stylized "chain" domain testing "deep exploration":**



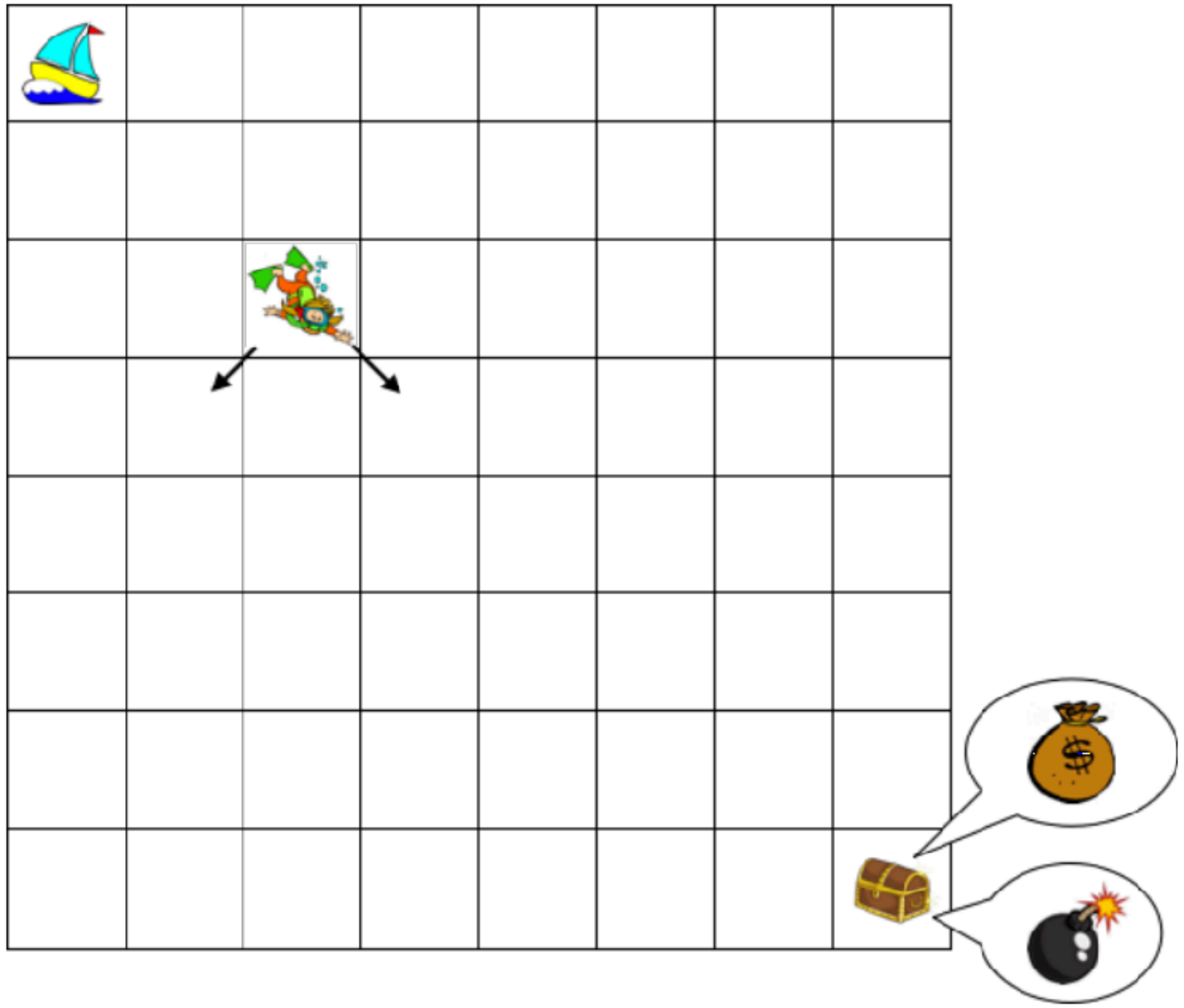
"Deep Sea" Exploration

- **Stylized "chain" domain testing "deep exploration":**
  - *State = N x N grid, observations 1-hot.*



"Deep Sea" Exploration

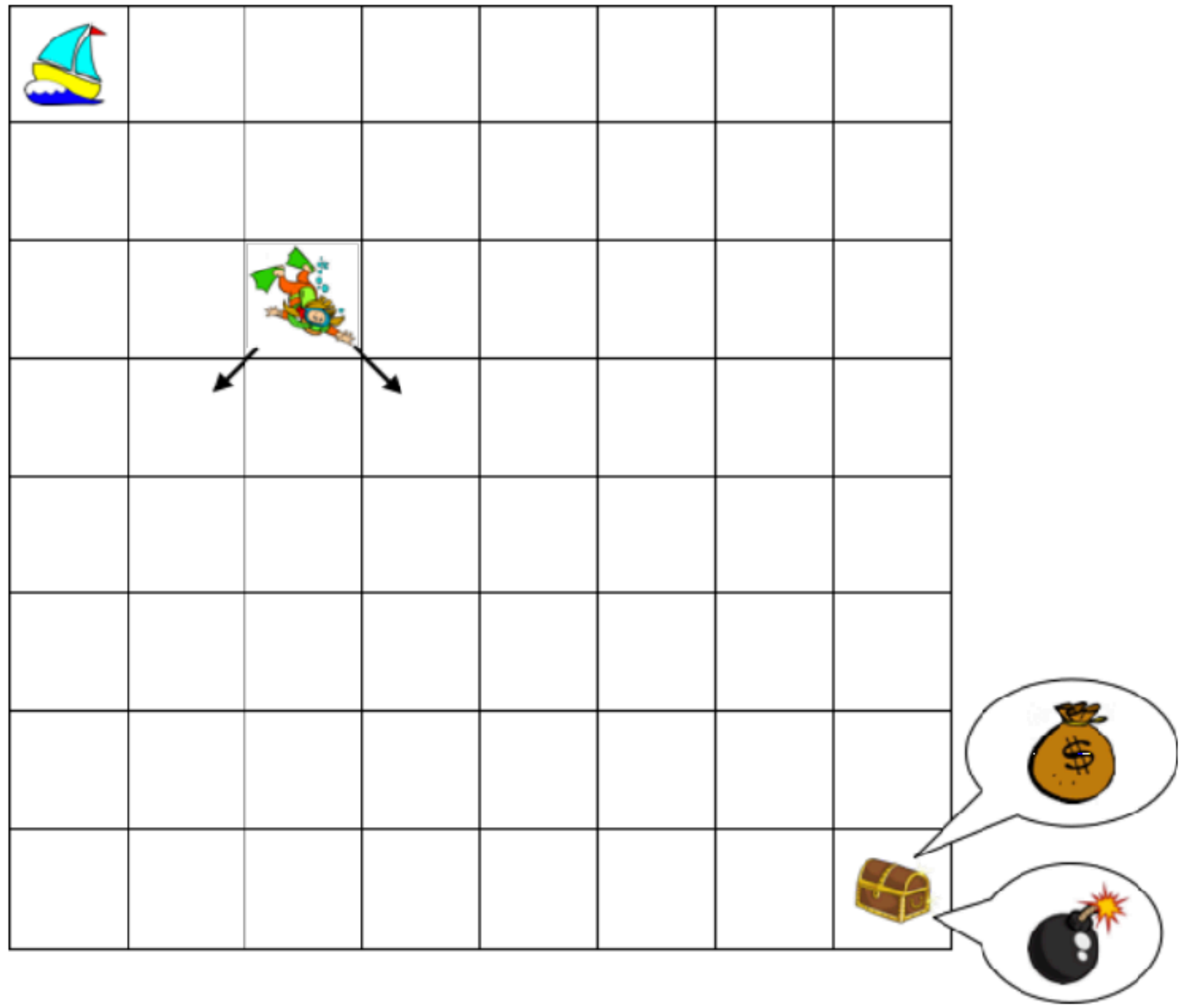
- **Stylized "chain" domain testing "deep exploration":**
  - *State = N x N grid, observations 1-hot.*
  - *Start in top left cell, fall one row each step.*



"Deep Sea" Exploration

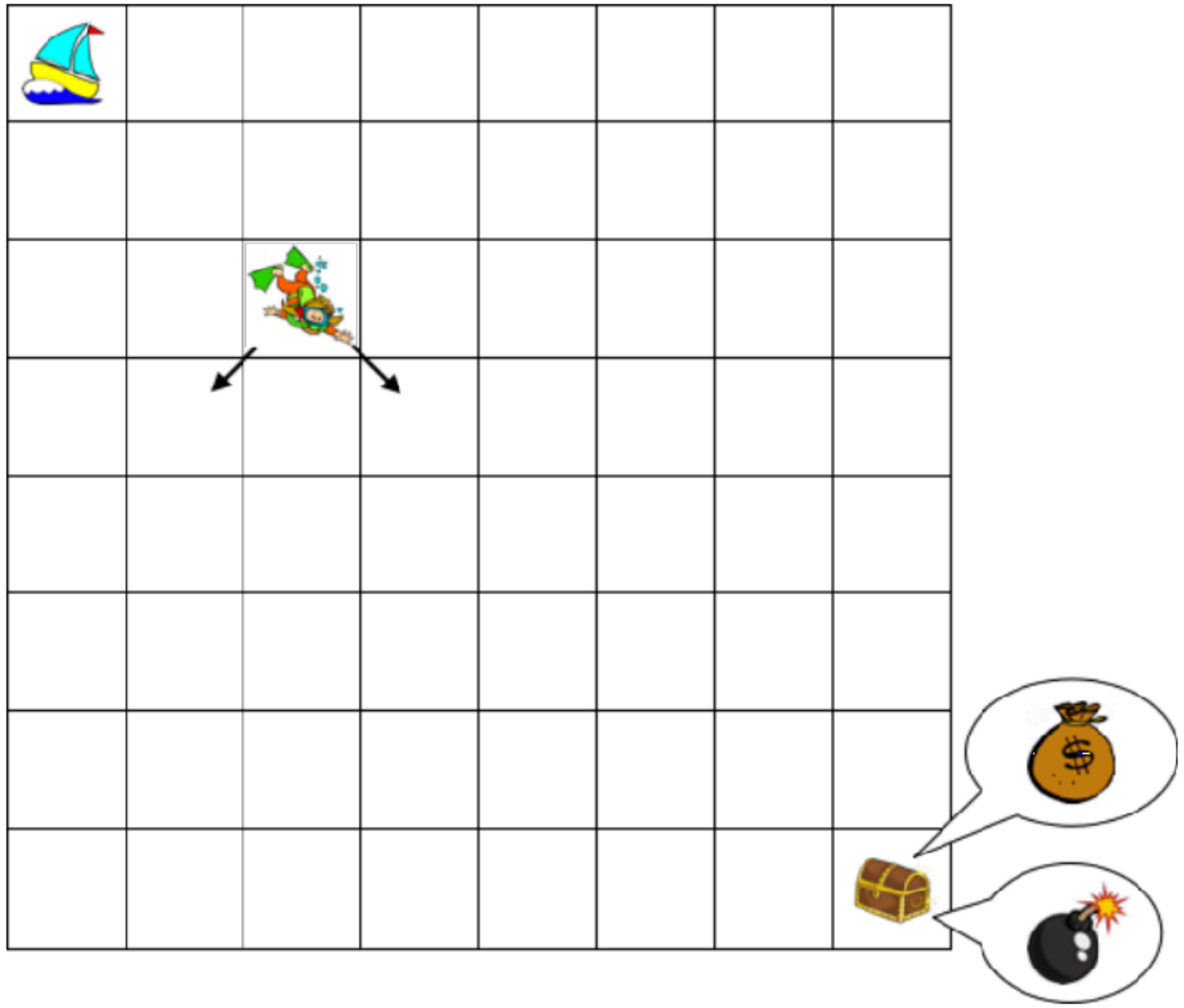
bit.ly/rpf_nips
@ianosband

- **Stylized "chain" domain testing "deep exploration":**
  - *State = N x N grid, observations 1-hot.*
  - *Start in top left cell, fall one row each step.*
  - *Actions {0,1} map to left/right in each cell.*



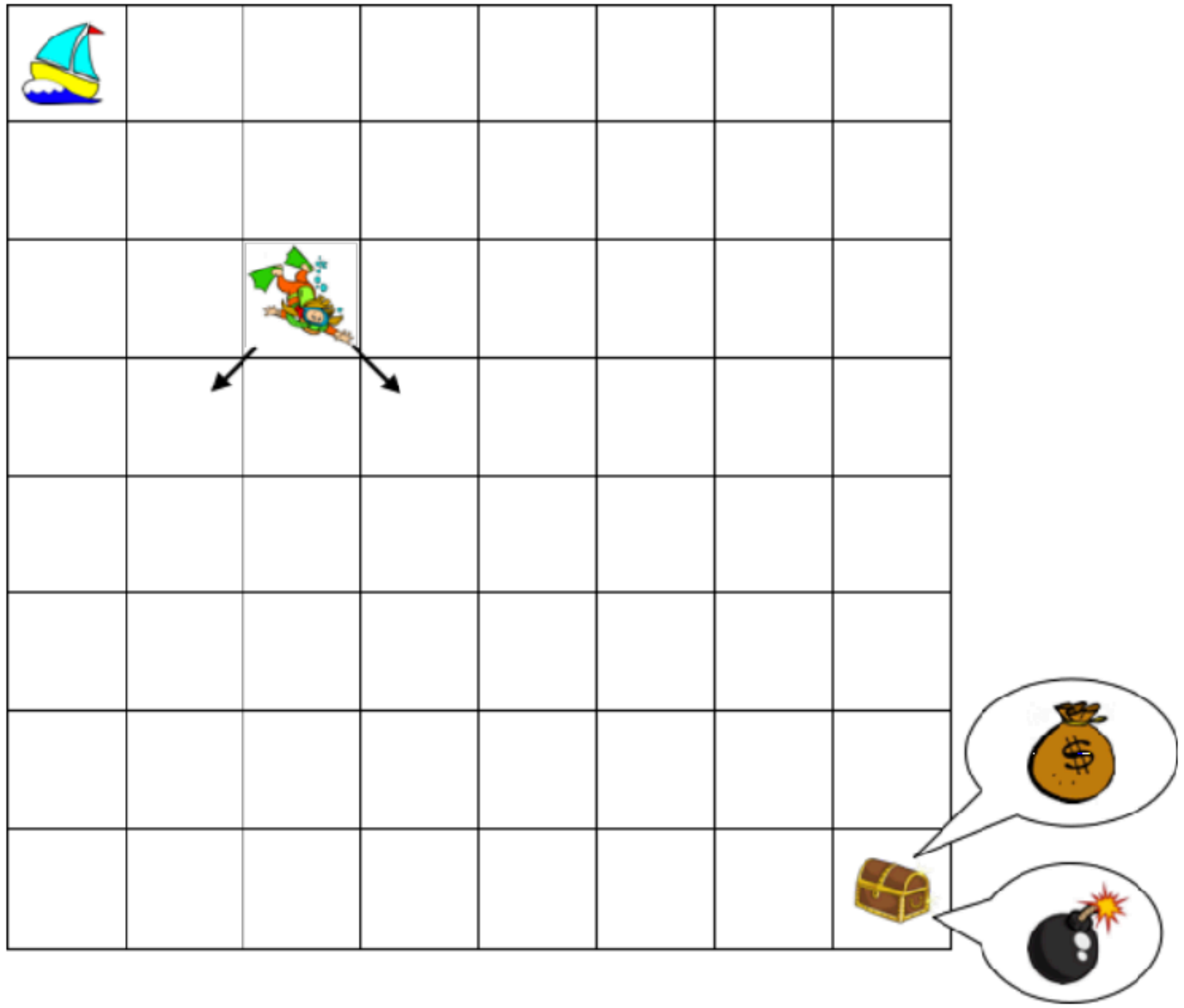
"Deep Sea" Exploration

- **Stylized "chain" domain testing "deep exploration":**
  - *State = N x N grid, observations 1-hot.*
  - *Start in top left cell, fall one row each step.*
  - *Actions {0,1} map to left/right in each cell.*
  - ***"left" has reward = 0**, **"right" has reward = -0.1/N***



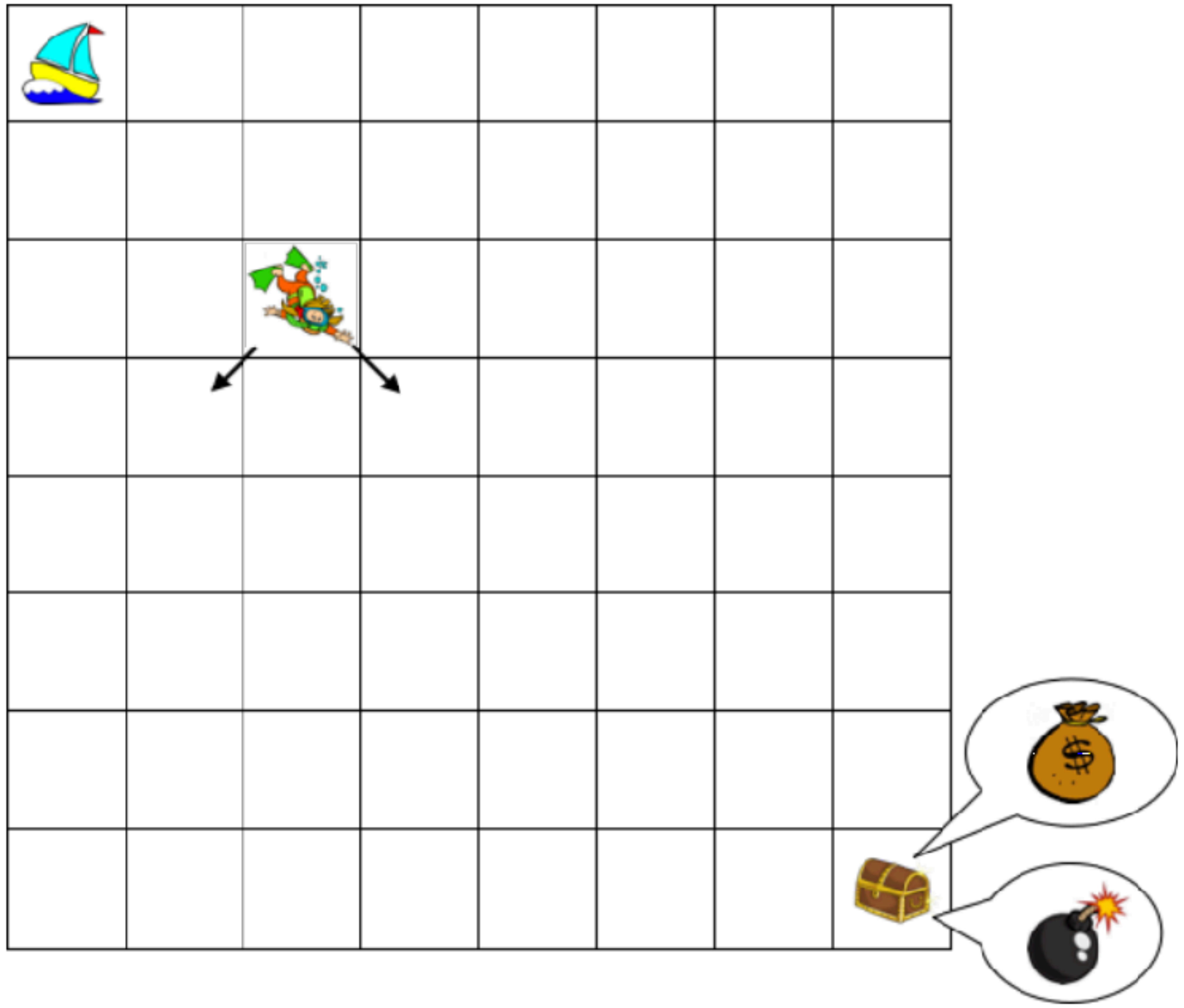
"Deep Sea" Exploration

- **Stylized "chain" domain testing "deep exploration":**
  - *State = N x N grid, observations 1-hot.*
  - *Start in top left cell, fall one row each step.*
  - *Actions {0,1} map to left/right in each cell.*
  - ***"left" has reward = 0**, **"right" has reward = -0.1/N***
  - ***… but if you make it to bottom right you get +1.***



"Deep Sea" Exploration

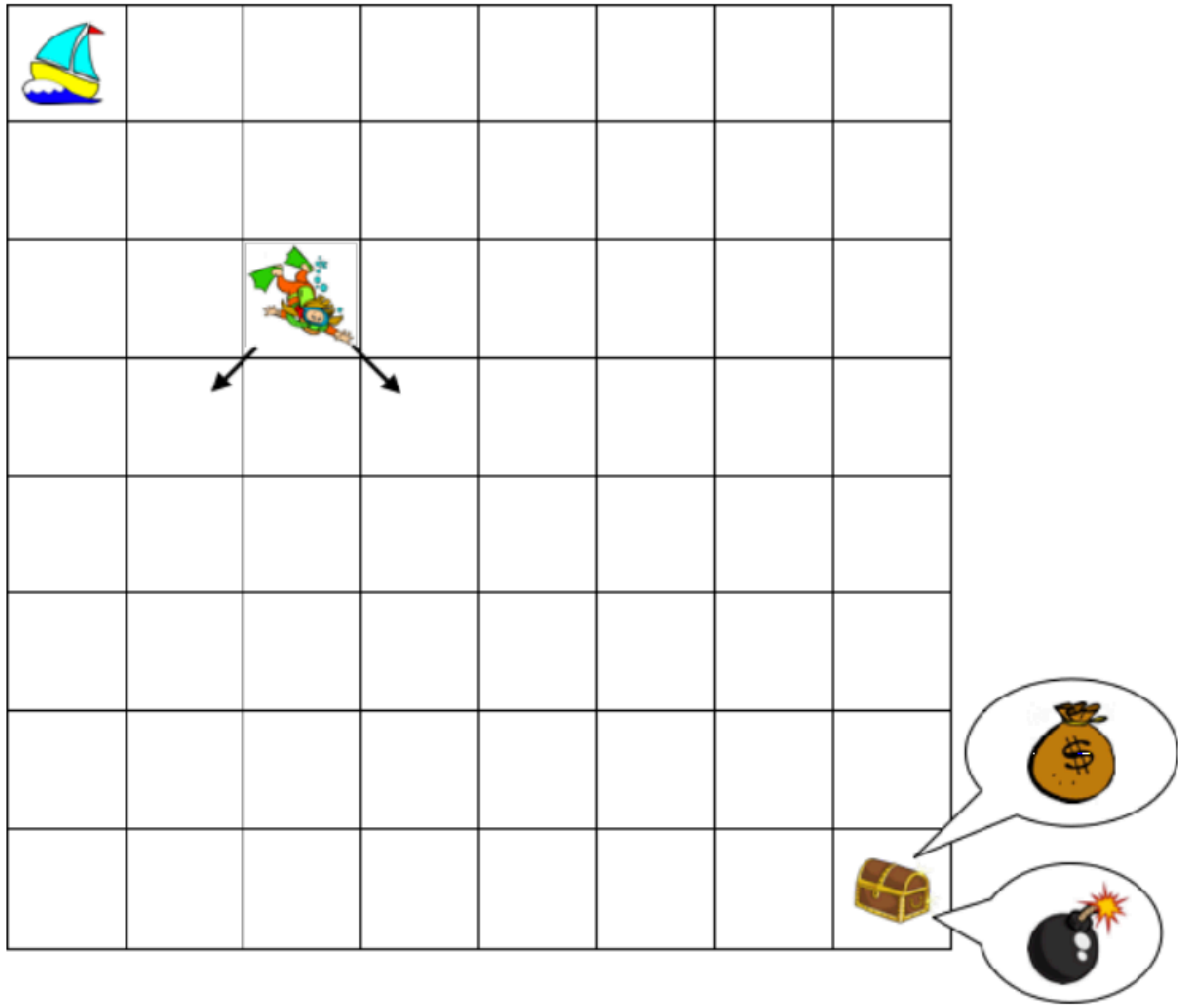
- **Stylized "chain" domain testing "deep exploration":**
  - *State = N x N grid, observations 1-hot.*
  - *Start in top left cell, fall one row each step.*
  - *Actions {0,1} map to left/right in each cell.*
  - ***"left" has reward = 0, "right" has reward = -0.1/N***
  - ***… but if you make it to bottom right you get +1.***



"Deep Sea" Exploration

- **Stylized "chain" domain testing "deep exploration":**
  - *State = N x N grid, observations 1-hot.*
  - *Start in top left cell, fall one row each step.*
  - *Actions {0,1} map to left/right in each cell.*
  - ***"left" has reward = 0**, **"right" has reward = -0.1/N***
  - ***… but if you make it to bottom right you get +1.***

- Only one policy (out of more than 2^N) has positive return.

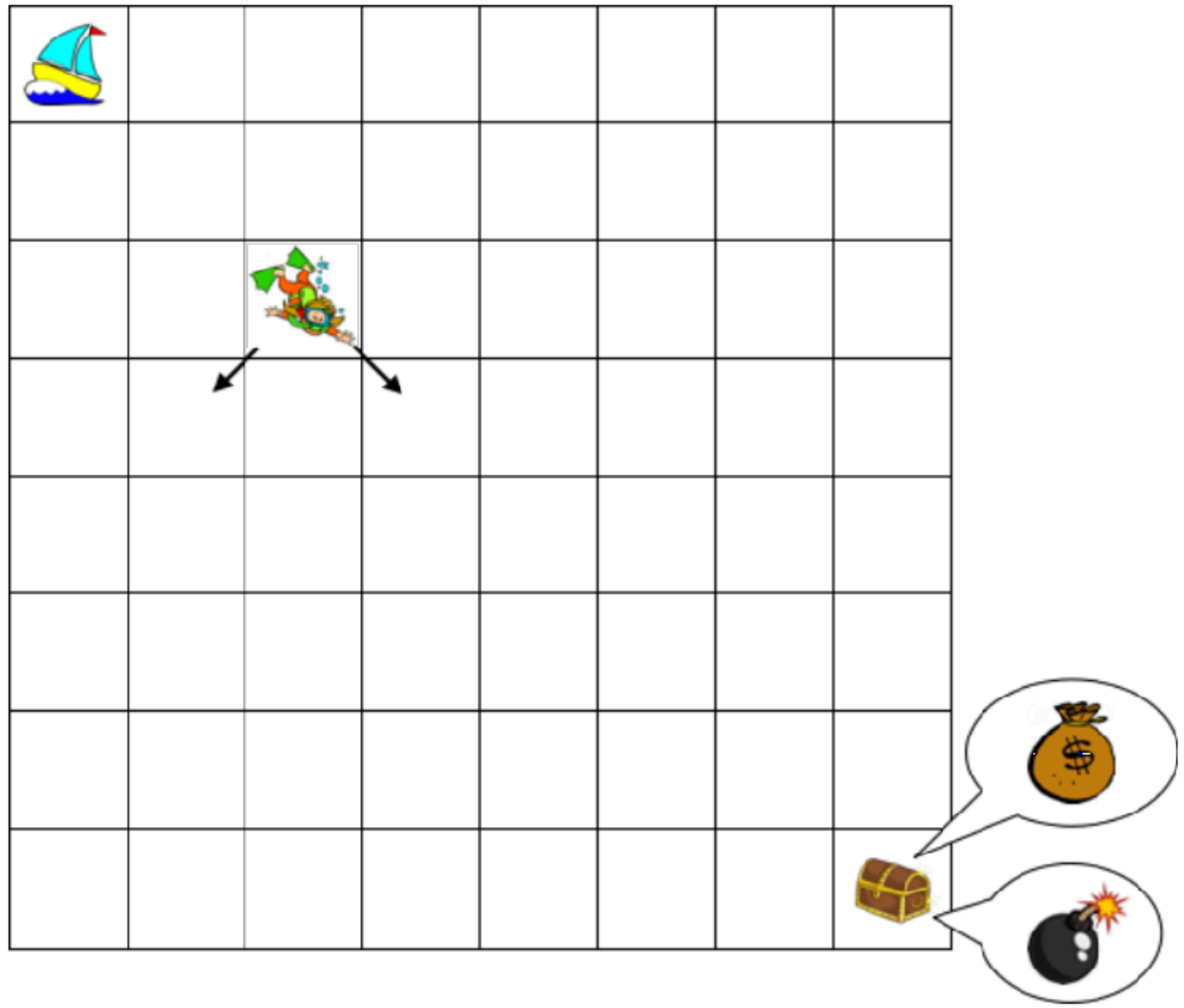


“Deep Sea” Exploration

- **Stylized “chain” domain testing “deep exploration”:**
  - *State = N x N grid, observations 1-hot.*
  - *Start in top left cell, fall one row each step.*
  - *Actions {0,1} map to left/right in each cell.*
  - ***“left” has reward = 0**, **“right” has reward = -0.1/N***
  - ***… but if you make it to bottom right you get +1.***

- Only one policy (out of more than 2^N) has positive return.

- **ε-greedy / Boltzmann / policy gradient / are useless.**



“Deep Sea” Exploration

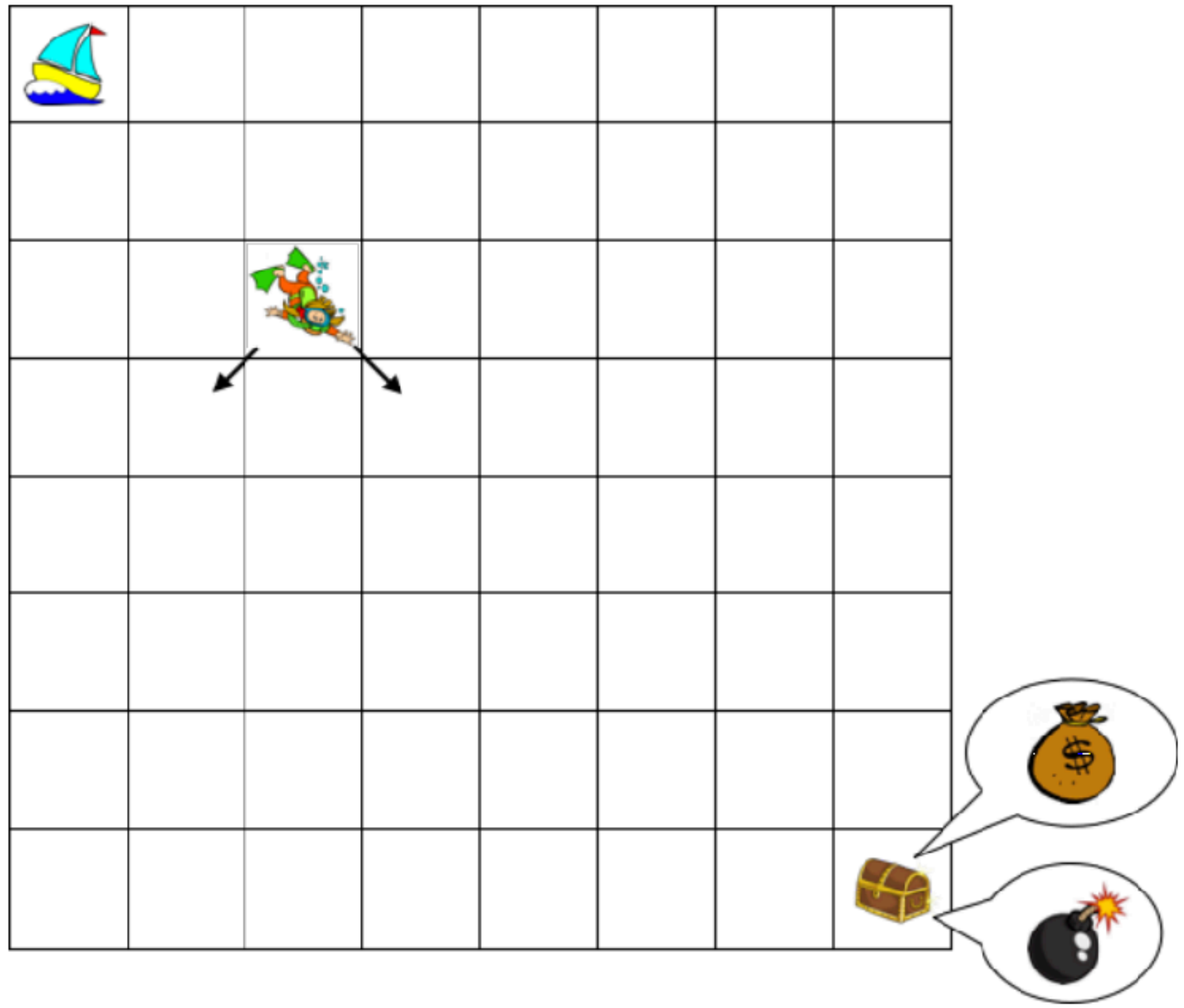
- **Stylized “chain” domain testing “deep exploration”:**
  - *State = N x N grid, observations 1-hot.*
  - *Start in top left cell, fall one row each step.*
  - *Actions {0,1} map to left/right in each cell.*
  - ***“left” has reward = 0**, **“right” has reward = -0.1/N***
  - ***… but if you make it to bottom right you get +1.***

- Only one policy (out of more than 2^N) has positive return.

- **ε-greedy / Boltzmann / policy gradient / are useless.**

- **Algorithms with deep exploration can learn fast!**

  [1] “Deep Exploration via Randomized Value Functions”



Visualize BootDQN+prior exploration.

bit.ly/rpf_nips
@ianosband

Visualize BootDQN+prior exploration.

- Compare **DQN+ε-greedy** vs **BootDQN+prior**.

Visualize BootDQN+prior exploration.

- Compare **DQN+ε-greedy** vs **BootDQN+prior**.
- Define ensemble average: $\frac{1}{K}\sum_{k=1}^{K}\max_{\alpha} Q_k(s, \alpha)$

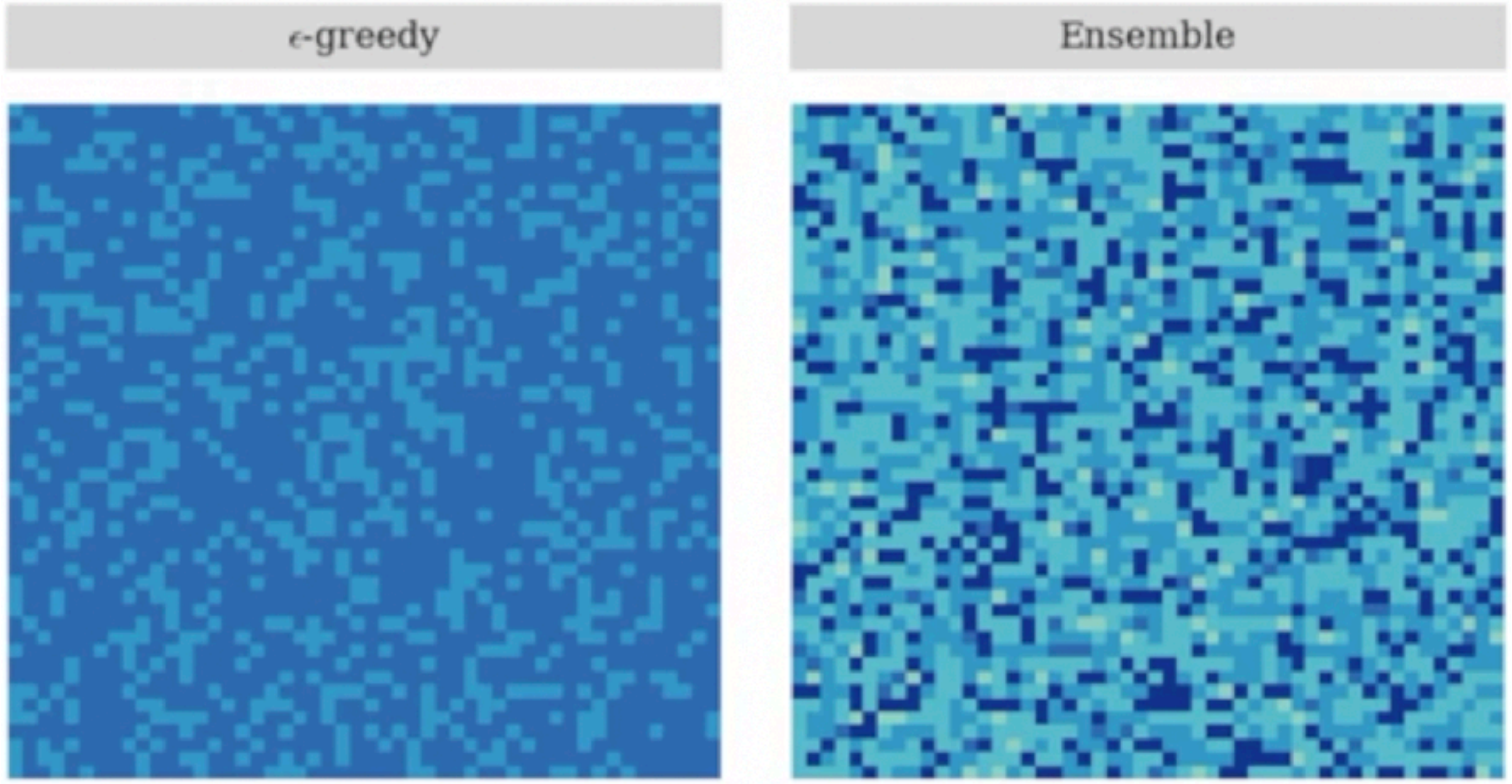
Visualize BootDQN+prior exploration.

- Compare **DQN+ε-greedy** vs **BootDQN+prior**.
- Define ensemble average: $\frac{1}{K}\sum_{k=1}^{K} \max_{\alpha} Q_k(s, \alpha)$
- Heat map shows estimated value of each state.

Visualize BootDQN+prior exploration.

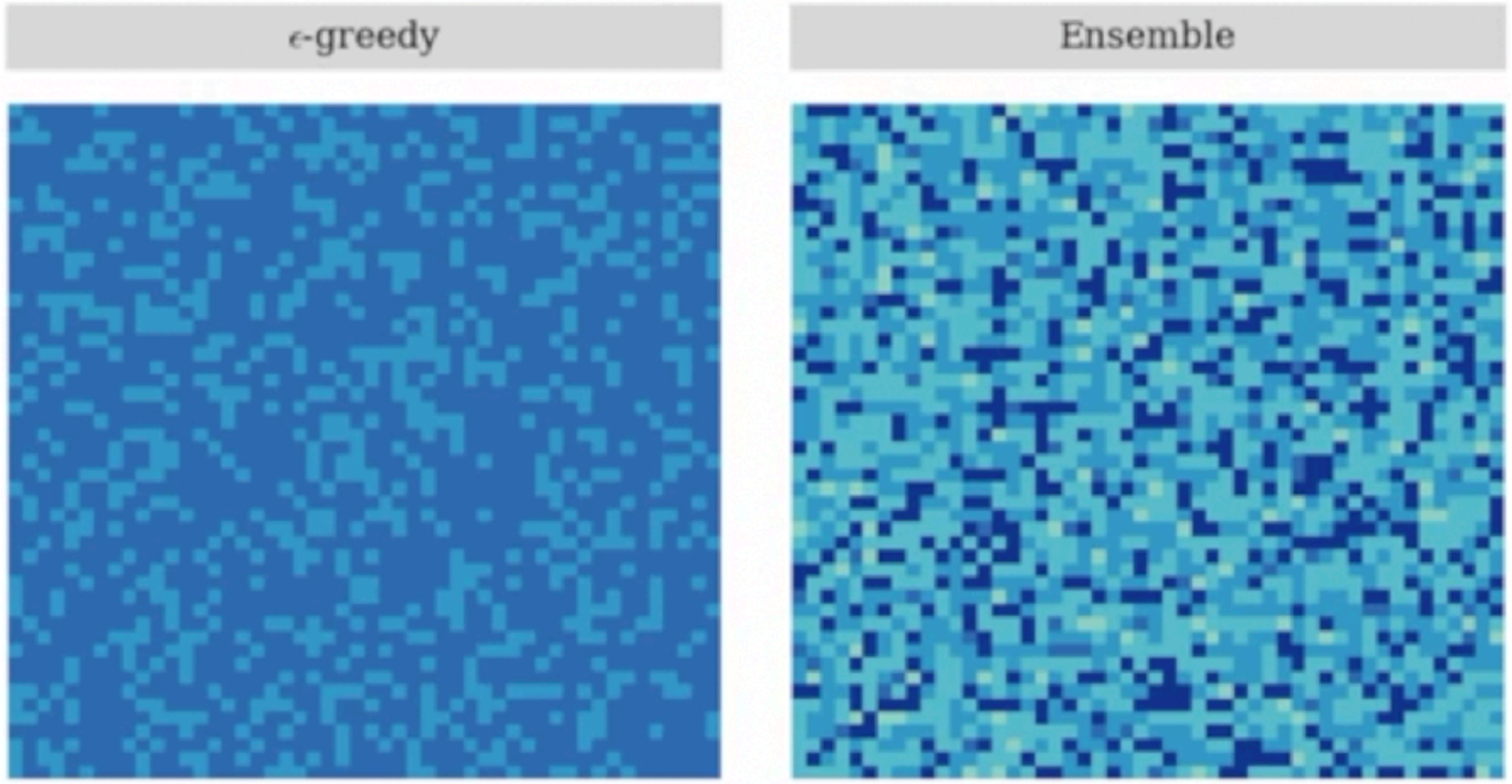
bit.ly/rpf_nips
@ianosband

- Compare **DQN+ε-greedy** vs **BootDQN+prior**.
- Define ensemble average: $\frac{1}{K}\sum_{k=1}^{K} \max_{\alpha} Q_k(s, \alpha)$
- Heat map shows estimated value of each state.



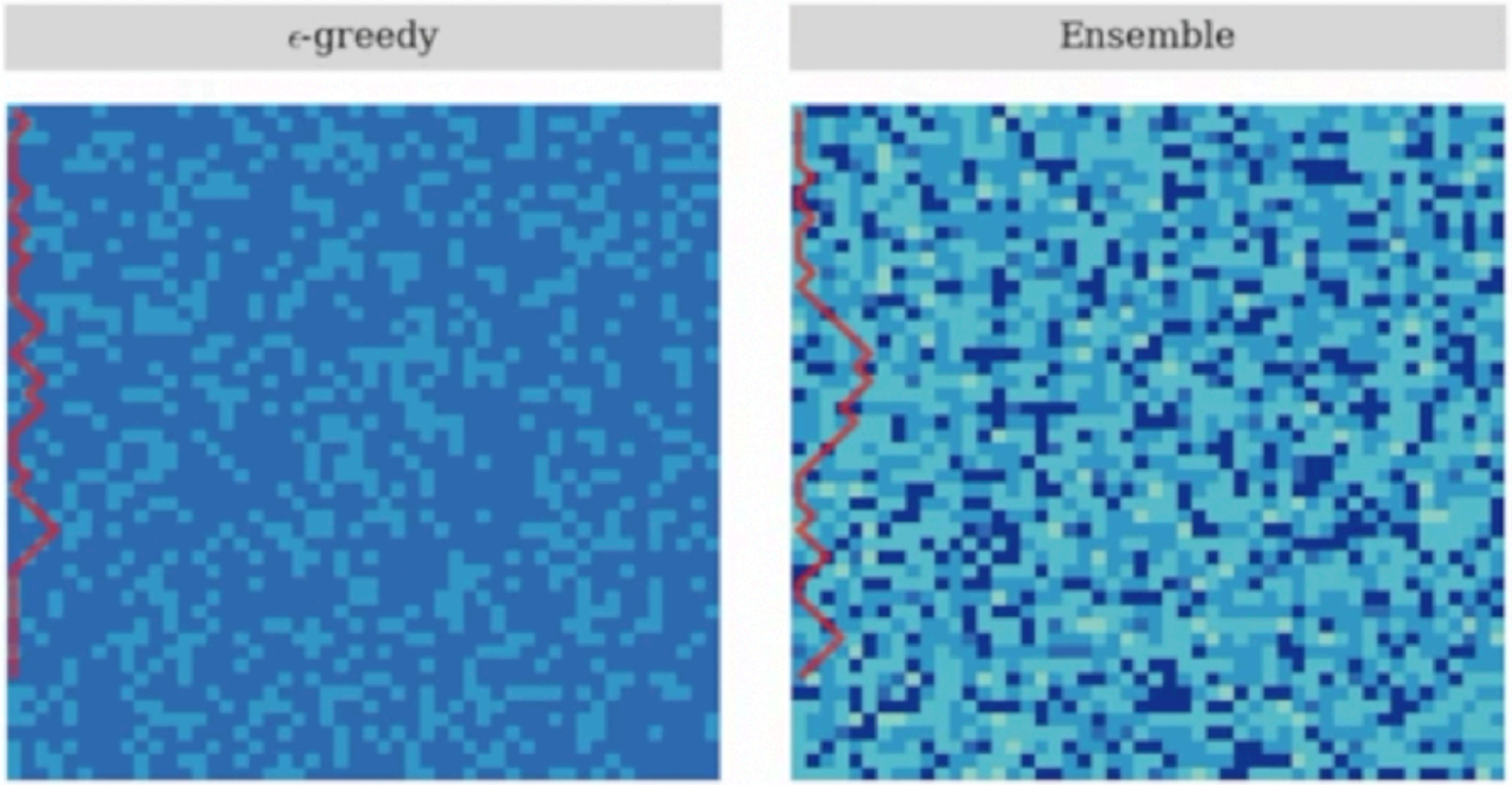
Visualize BootDQN+prior exploration.

- Compare **DQN+ε-greedy** vs **BootDQN+prior**.
- Define ensemble average: $\frac{1}{K}\sum_{k=1}^{K} \max_{\alpha} Q_k(s, \alpha)$
- Heat map shows estimated value of each state.
- **Red line** shows exploration path taken by agent.



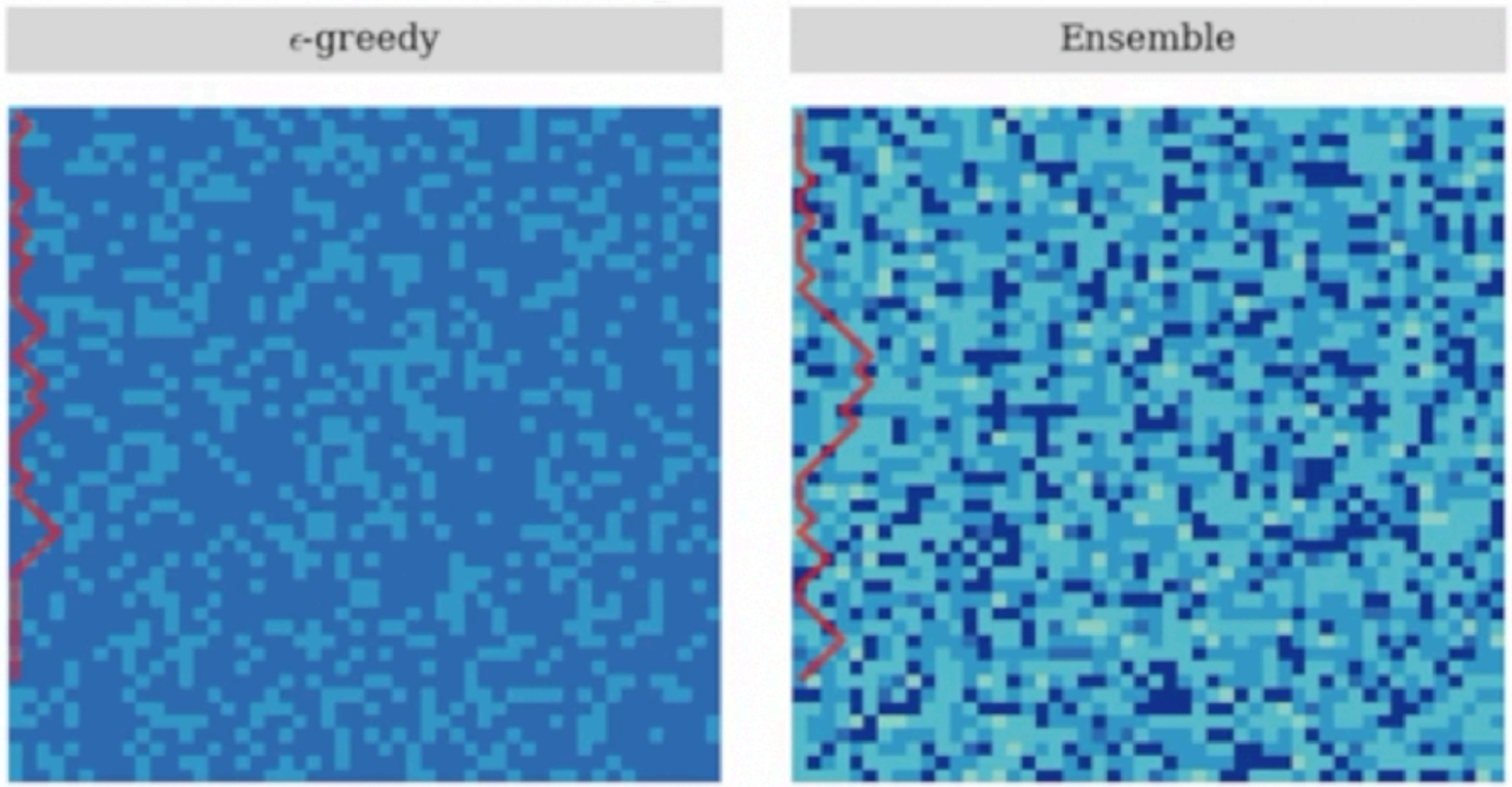
Visualize BootDQN+prior exploration.

- Compare **DQN+ε-greedy** vs **BootDQN+prior**.
- Define ensemble average: $\frac{1}{K}\sum_{k=1}^{K}\max_{\alpha} Q_k(s,\alpha)$
- Heat map shows estimated value of each state.
- **Red line** shows exploration path taken by agent.



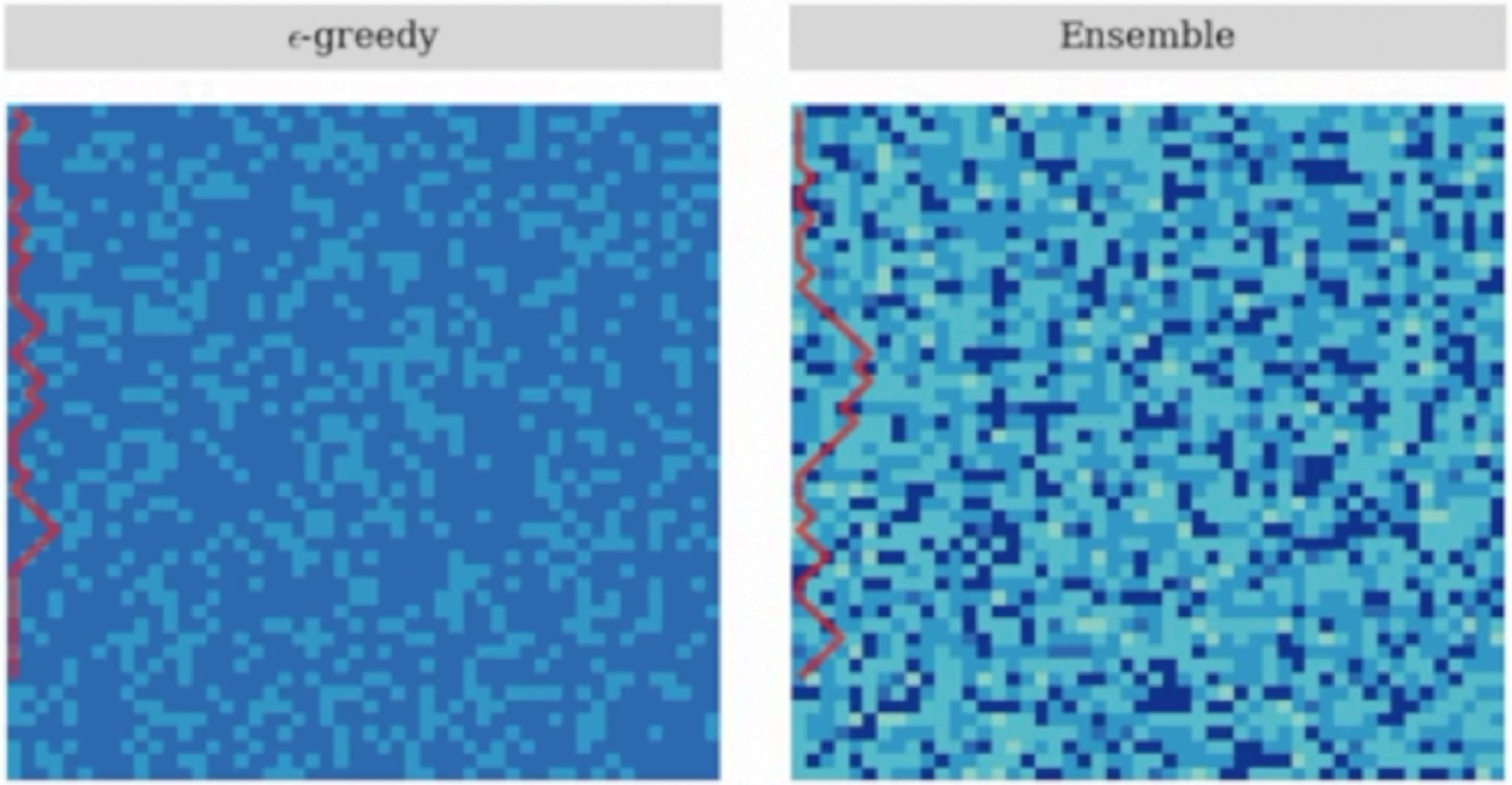
Visualize BootDQN+prior exploration.

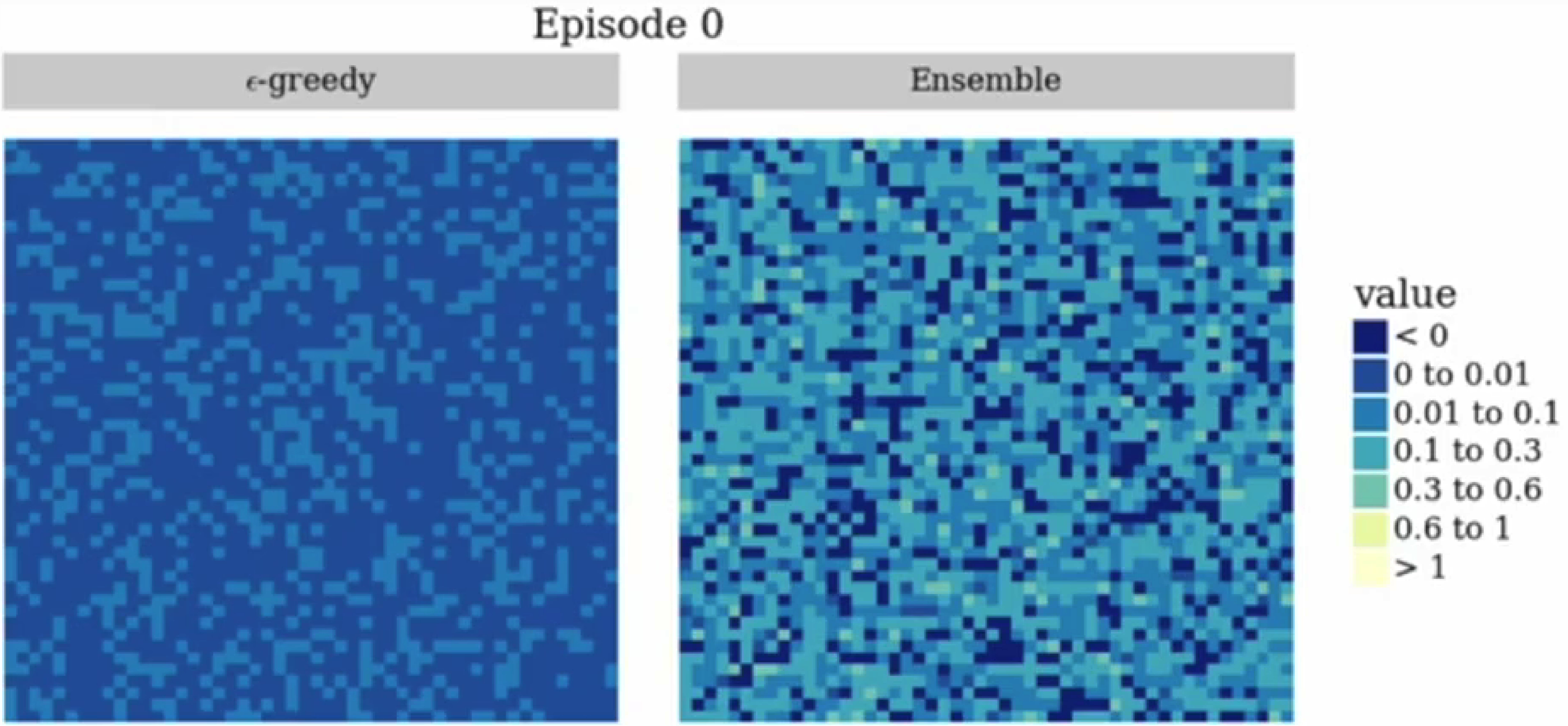
- Compare **DQN+ε-greedy** vs **BootDQN+prior**.
- Define ensemble average: $\frac{1}{K}\sum_{k=1}^{K}\max_{\alpha} Q_k(s,\alpha)$
- Heat map shows estimated value of each state.
- **Red line** shows exploration path taken by agent.
- **DQN+ε-greedy gets stuck on the left, gives up.**



Visualize BootDQN+prior exploration.

- Compare **DQN+ε-greedy** vs **BootDQN+prior**.
- Define ensemble average: $\frac{1}{K}\sum_{k=1}^{K}\max_{\alpha} Q_k(s,\alpha)$
- Heat map shows estimated value of each state.
- **Red line** shows exploration path taken by agent.
- **DQN+ε-greedy gets stuck on the left, gives up.**
- **BootDQN+prior hopes something is out there, keeps exploring potentially-rewarding states… learns fast!**




Episode 0
ϵ-greedy
Ensemble
value
< 0
0 to 0.01
0.01 to 0.1
0.1 to 0.3
0.3 to 0.6
0.6 to 1
> 1

Come visit our poster!

Ian Osband, John Aslanides, Albin Cassirer

Come visit our poster!

Ian Osband, John Aslanides, Albin Cassirer

Blog post
bit.ly/rpf_nips

Come visit our poster!

Ian Osband, John Aslanides, Albin Cassirer

Blog post
bit.ly/rpf_nips

Montezuma's Revenge!

Come visit our poster!

Ian Osband, John Aslanides, Albin Cassirer

Blog post
bit.ly/rpf_nips

Montezuma's Revenge!

Demo code
bit.ly/rpf_nips