OCTOBER 2025

TEXAS
The University of Texas at Austin

SIMPLE POLICY OPTIMIZATION

Ajay Yadav

Contents:

- Overview…….. (3)
- Background ……..…….. (5)
- TRPO……..……..…….. (8)
- PPO……..……..……(12)
- SPO……..……..(15)
- Experiments……..……..……..(20)
- Conclusion……..(26)

TEXAS
The University of Texas at Austin

OVERVIEW

What is SPO?

SPO is an algorithm that addresses a significant flaw in PPO, one of today's most widely used reinforcement learning methods.

The Problem it Solves: PPO often becomes unstable when training the large, deep neural networks required for complex tasks.

SPO's Solution: It makes one small but powerful change to PPO's objective function to guarantee more stable and reliable training.

The "Best of Both Worlds": It achieves the rock-solid stability of older, complex methods (like TRPO) while keeping the speed and simplicity of PPO.

TEXAS
The University of Texas at Austin

BACKGROUND

The Goal of Policy Gradient RL

Goal:

- In Reinforcement Learning, we want an agent to learn a policy π that maximizes its cumulative reward.

- Policy Gradient methods do this by directly adjusting the policy's parameters (θ) in the direction that promises more reward.

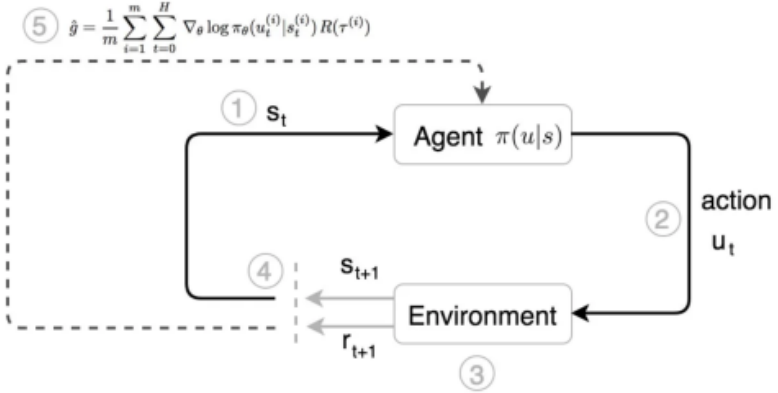
Challenge: How big a step should we take?

- **Too Small:** Learning is extremely slow and sample-inefficient.

- **Too Large:** A single bad update can cause the policy to completely collapse, a catastrophic failure from which it may never recover.

The key is to find the "Goldilocks zone":

- Make the biggest possible step without destabilizing the policy.



Two OG Algorithms

TRPO (Trust Region Policy Optimization)

- **Strength:** Guarantees stable, monotonic improvement by mathematically defining a "trust region" and keeping updates within it. It's very reliable.

- **Weakness:** Uses complex and computationally expensive second-order optimization. It's difficult to implement and too slow for many large-scale problems.

PPO (Proximal Policy Optimization)

- **Strength:** Drastically simplifies TRPO using only first-order optimization. It's fast, simple to implement, and has become the default choice for many RL tasks.

- **Weakness:** Its core mechanism—ratio clipping—is a heuristic that *often fails* to keep the policy within the trust region, leading to instability.

TEXAS
The University of Texas at Austin

TRPO

TRPO

The main goal of TRPO is to answer the question: ***"How can I improve my current policy (π) to get a better one (π') without risking a total collapse in performance?"***

Answer: The improvement in performance between our old policy and the new one is equal to the average advantage of the actions the new policy takes.

$$\eta(\tilde{\pi}) - \eta(\pi) = \frac{1}{1-\gamma}\mathbb{E}_{s\sim\rho_{\tilde{\pi}}(\cdot),a\sim\tilde{\pi}(\cdot|s)}[A_{\pi}(s,a)]$$

$\eta(\pi)$: *Total expected Reward for policy π*
π': *The new policy*
π: *Our current policy*
$A_{\pi}(s, a)$: Advantage function

In Simple Words: If our new policy (π') consistently chooses actions that are better than the old policy's average action in those same situations, our overall score will go up.

The Problem: Notice the subscript $s \sim \rho\tilde{}\pi(\cdot)$. This means the average is calculated over the states visited by the new policy. We don't have data for that yet! We can't use this equation directly to make an update.

TRPO Fix: Find a safe lower bound

This is the core trick of TRPO. Since we can't calculate the exact improvement, we find a guaranteed minimum improvement instead, called a “performance lower bound.”

$$\eta(\tilde{\pi}) - \eta(\pi) \geq \frac{1}{1-\gamma}\mathbb{E}_{s\sim\rho_\pi(\cdot),a\sim\pi(\cdot|s)}\left[\frac{\tilde{\pi}(a|s)}{\pi(a|s)} \cdot A_\pi(s,a)\right] - \text{Penalty Term}$$

What it says: The performance improvement is, at minimum, equal to the expected advantage, but we have to subtract a penalty if the new policy becomes too different from the old one.

In Simple Words: We can estimate the improvement using data from our old policy (notice the subscript changed to $s \sim \rho_\pi$), but we must be careful.

If we change the policy too much, this estimate becomes unreliable. The penalty term formalizes this “be careful” idea.

$\frac{\tilde{\pi}(a|s)}{\pi(a|s)}$ This is an importance sampling ratio. It reweights the advantage to account for using data from the old policy.

$$\text{Sampling Ratio} = \frac{Probability\ new\ policy\ takes\ action\ a}{Probability\ old\ policy\ takes\ action\ a}$$

Penalty Term : is KL Divergence D_{KL}

TRPO: Final Optimization

Putting everything together we get

$$\max_{\theta} \mathbb{E}_{(s_t,a_t)\sim\pi_{\theta_{\text{old}}}} \left[\frac{\pi_\theta(a_t|s_t)}{\pi_{\theta_{\text{old}}}(a_t|s_t)} \cdot \hat{A}(s_t,a_t) \right] \quad \text{s.t.} \quad \mathbb{E}[D_{KL}(\pi_{\theta_{\text{old}}}||\pi_\theta)] \leq \delta$$

What it says: Our goal is to maximize the estimated advantage (the "surrogate objective")...but with a very important constraint.

In Simple Words: "Make the policy as good as you can, BUT do not let the new policy deviate from the old one by more than a small amount, δ."

- θ: The parameters of our neural network that defines the policy.
- $\pi_{\theta_{\text{old}}}$: The old policy (before the update).
- π_θ: The new policy (that we are trying to find).
- $\hat{A}$: An *estimate* of the advantage function from our collected data.
- $D_{KL}(\pi_{\theta_{\text{old}}}||\pi_\theta) \leq \delta$: This is the **Trust Region Constraint**, the mathematical rule for "don't change the policy too much."

TEXAS
The University of Texas at Austin

PPO

PPO

Proximal Policy Optimization (PPO) was designed to get the same benefits as TRPO—stable, reliable policy updates—but without the complex and computationally expensive second-order optimization.
The core question PPO answers is: ***"Can we get trust region benefits using only first-order gradients, making the algorithm simpler and faster?"***

$$J_{\text{clip}}(\theta) = \mathbb{E}_{(s_t,a_t)\sim\pi_{\theta_{\text{old}}}}\left[\min\left(r_t(\theta)\cdot\hat{A}_t, \text{clip}(r_t(\theta), 1-\epsilon, 1+\epsilon)\cdot\hat{A}_t\right)\right]$$

$r_t(\theta) = \frac{\pi_\theta(a_t|s_t)}{\pi_{\theta_{\text{old}}}(a_t|s_t)}$ is the standard importance sampling ratio.

$$\text{Sampling Ratio} = \frac{Probability\ new\ policy\ takes\ action\ a}{Probability\ old\ policy\ takes\ action\ a}$$

$\hat{A}_t$ is the estimated advantage for taking action a_t in state s_t.

ϵ is a small hyperparameter (e.g., 0.2) that defines the size of the trust region.

$\text{clip}(r_t(\theta), 1-\epsilon, 1+\epsilon)$: This function constrains the ratio $r_t(\theta)$ to stay within the range $[1-\epsilon, 1+\epsilon]$.

The min function is the key to the algorithm. It takes a minimum of two terms: the normal policy objective and the clipped version. This creates a pessimistic, lower-bound objective.

PPO: How the Clipping works

The clipping has a different effect depending on whether the advantage is positive or negative.

$$\nabla_\theta J_{\text{clip}}(\theta) = \begin{cases} \nabla_\theta r_t(\theta) \cdot \hat{A}_t, & \hat{A}_t > 0, r_t(\theta) \leq 1+\epsilon; \\ \nabla_\theta r_t(\theta) \cdot \hat{A}_t, & \hat{A}_t < 0, r_t(\theta) \geq 1-\epsilon; \\ 0, & \text{otherwise.} \end{cases}$$

- **When Advantage is Positive $A'_t > 0$:** The agent wants to increase the probability of taking this action.
- **When Advantage is Negative $A'_t < 0$:** The agent wants to decrease the probability of taking this ``bad'' action.

But there is a problem:
While simple and often effective, the clipping mechanism is a heuristic, not a guarantee.

- **Zero Gradient Problem:** The biggest issue is that once the ratio $\mathbf{r_t(\theta)}$ moves outside the [1 − ϵ, 1 + ϵ] clipping range, the objective function becomes flat.

- **The Consequence:** A flat objective means the gradient for that data point becomes zero. The algorithm loses its “corrective signal”—it no longer receives a gradient to push the policy back inside the trust region. This can lead to instability, especially when training complex models where large, uncontrolled updates are more likely.

TEXAS
The University of Texas at Austin

SPO

SPO

Simple Policy Optimization (SPO) was created to directly solve the **“zero gradient”** problem that makes PPO unstable. SPO’s goal is to keep the simplicity and efficiency of PPO but enforce the trust region constraint in a more reliable and mathematically sound way. The core question SPO answers is: **“Can we design a new objective function that provides a continuous corrective signal to keep the policy within the trust region?”**

$$J_{\mathrm{SPO}}(\theta) = \mathbb{E}_{(s_t,a_t)\sim\pi_{\theta_{\mathrm{old}}}}\left\{ r_t(\theta)\cdot\hat{A}_t - \frac{|\hat{A}_t|}{2\epsilon}\cdot[r_t(\theta)-1]^2 \right\}$$

$r_t(\theta) \cdot A'_t$: “Surrogate Objective”. This term drives the policy to take actions that have a higher estimated advantage. It’s the “performance-seeking” part of the function

$-\frac{|\hat{A}_t|}{2\epsilon}\cdot[r_t(\theta)-1]^2$: This is the novel **SPO penalty**, and it is the core contribution of the algorithm.

SPO: Penalty Term

$$-\frac{|\hat{A}_t|}{2\epsilon} \cdot [r_t(\theta) - 1]^2$$

- **$[r_t(\theta) - 1]^2$** is a quadratic penalty. Unlike PPO's hard clip, this creates a smooth penalty that increases progressively in size as the ratio **$r_t(\theta)$** moves away from 1.
- Because it is a smooth function, its gradient is never zero (unless the ratio is perfectly on target).
- This means there is always a corrective force that gently pulls the policy ratio back towards the trust region boundary, preventing it from drifting too far away.

SPO: Why it works?

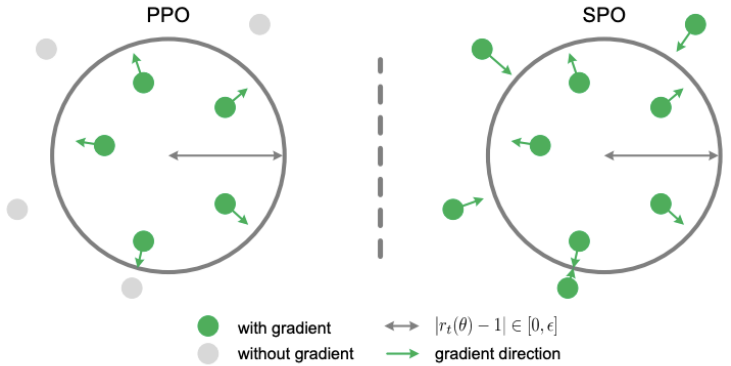
The “ϵ-aligned” Objective: The paper provides a formal property to explain why the SPO objective is so effective. it is “ϵ-aligned.”

$$r^* = 1 + \text{sign}(A) \cdot \epsilon$$

The formula defines the **optimal probability ratio (r*)**. It's the "perfect" or "optimal" value for the probability ratio that will give the best policy improvement without leaving the trust region.

If the advantage is positive **(A > 0)**, the objective is maximized when the ratio is at the upper bound, **1 + ϵ**. The penalty gently pulls it back if it tries to exceed this

If the advantage is negative **(A < 0),** the objective is maximized when the ratio is at the lower bound, **1 − ϵ**. The penalty again provides a corrective pull if it tries to go lower.



SPO: Algorithm

Algorithm 1 Simple Policy Optimization (SPO)

Initialize: Policy and value networks π_θ, V_ϕ, hyperparameter ϵ, value loss and policy entropy coefficients c_1, c_2
Output: Optimal policy network π_{θ^*}
while not converged **do**
  # Data collection
  Collect data $\mathcal{D} = \{(s_t, a_t, r_t)\}_{t=1}^N$ using the current policy network π_θ
  # The networks before updating
  $\pi_{\theta_{\text{old}}} \leftarrow \pi_\theta, \quad V_{\phi_{\text{old}}} \leftarrow V_\phi$
  # Estimate the advantage $\hat{A}(s_t, a_t)$ based on $V_{\phi_{\text{old}}}$
  Use GAE (Schulman et al., 2015b) technique to estimate the advantage $\hat{A}(s_t, a_t)$
  # Estimate the return $\hat{R}_t$
  $\hat{R}_t \leftarrow V_{\phi_{\text{old}}}(s_t) + \hat{A}(s_t, a_t)$
  for each training epoch **do**
    # Compute policy loss $\mathcal{L}_p$ (This is the only difference between SPO and PPO)
    $\mathcal{L}_p \leftarrow -\frac{1}{N}\sum_{t=1}^N \left\{ \frac{\pi_\theta(a_t|s_t)}{\pi_{\theta_{\text{old}}}(a_t|s_t)} \cdot \hat{A}(s_t, a_t) - \frac{|\hat{A}(s_t,a_t)|}{2\epsilon} \cdot \left[\frac{\pi_\theta(a_t|s_t)}{\pi_{\theta_{\text{old}}}(a_t|s_t)} - 1 \right]^2 \right\}$
    # Compute policy entropy $\mathcal{L}_e$ and value loss $\mathcal{L}_v$
    $\mathcal{L}_e \leftarrow \frac{1}{N}\sum_{t=1}^N \mathcal{H}(\pi_\theta(\cdot|s_t)), \quad \mathcal{L}_v \leftarrow \frac{1}{2N}\sum_{t=1}^N [V_\phi(s_t) - \hat{R}_t]^2$
    # Compute total loss $\mathcal{L}$
    $\mathcal{L} \leftarrow \mathcal{L}_p + c_1\mathcal{L}_v - c_2\mathcal{L}_e$
    # Update parameters θ and ϕ through backpropagation, λ_θ and λ_ϕ is the step sizes
    $\theta \leftarrow \theta - \lambda_\theta \nabla_\theta \mathcal{L}, \quad \phi \leftarrow \phi - \lambda_\phi \nabla_\phi \mathcal{L}$
  end for
end while

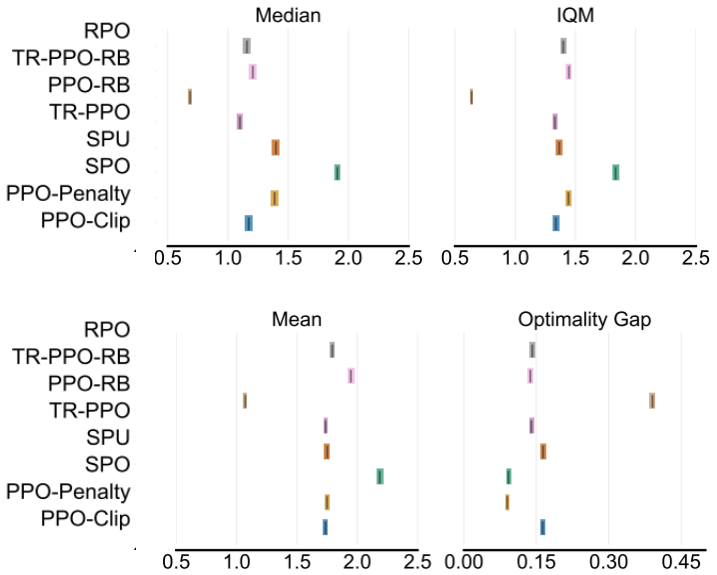
TEXAS
The University of Texas at Austin

EXPERIMENTS

Experiment 1: Benchmarking on MuJoCo

Goal: To compare SPO's performance against PPO and a suite of other modern policy gradient algorithms on standard continuous control benchmarks.
Setup: The algorithms were tested on six MuJoCo-v4 environments. Performance was aggregated and compared using several statistical metrics like Median, Interquartile Mean (IQM), and Optimality Gap.
Results:

- The paper shows in Figure 4 that SPO achieved the best performance across almost all metrics.
- It had a higher median, IQM, and mean score, and a lower optimality gap compared to PPO-Clip, PPO-Penalty, TR-PPO, and others.
- This demonstrates that even with a simple implementation, SPO is a highly competitive and robust algorithm for standard tasks.



Experiment 2: Scaling Policy Networks

Goal: To test the central hypothesis that SPO's stability allows it to successfully train a deep policy networks, a known failure point for PPO.
Setup:

- The researchers conducted two key sets of experiments:
- In MuJoCo, they increased the policy network depth from a standard 3 layers to a much deeper network, 7 layers, for both PPO and SPO.
- In Atari 2600, they replaced the default CNN encoder with a much larger ResNet-18 network.

Table 2. Detailed hyperparameters used in SPO.

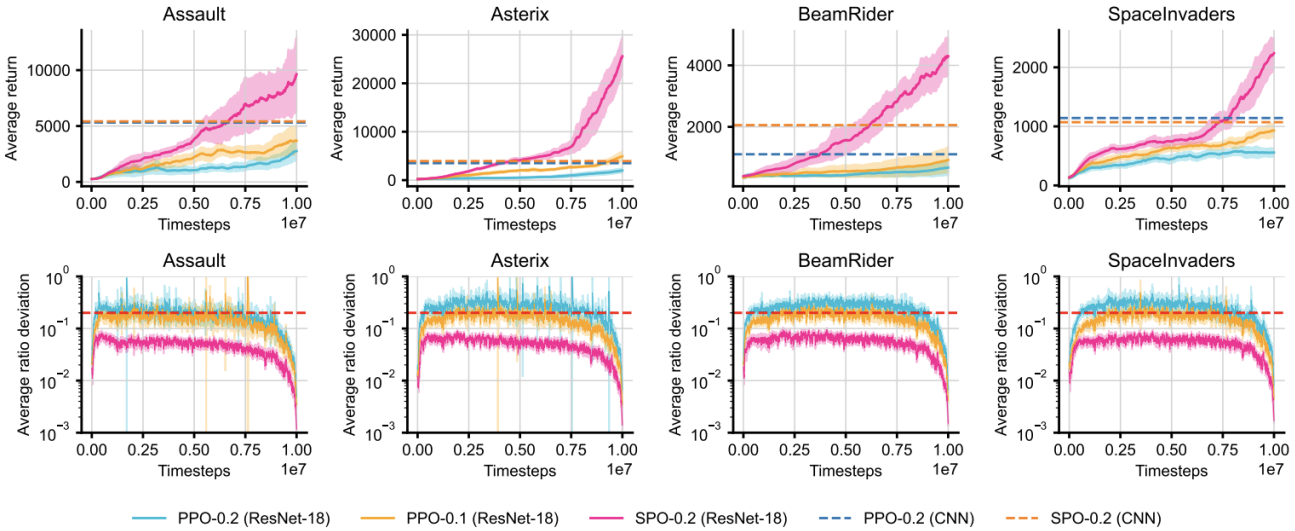
Hyperparameters	Atari 2600 (Bellemare et al., 2013)	MuJoCo (Todorov et al., 2012)
Number of workers	8	8
Horizon	128	256
Learning rate	0.00025	0.0003
Learning rate decay	Linear	Linear
Optimizer	Adam	Adam
Total steps	10M	10M
Batch size	1024	2048
Update epochs	4	10
Mini-batches	4	4
Mini-batch size	256	512
GAE parameter λ	0.95	0.95
Discount factor γ	0.99	0.99
Value loss coefficient c_1	0.5	0.5
Entropy loss coefficient c_2	0.01	0.0
Probability ratio hyperparameter ϵ	0.2	0.2

Experiment 2: Scaling Policy Networks

• PPO performance collapsed when the network depth was increased to 7 layers; PPO's performance catastrophically dropped in most MuJoCo environments. **Table 1** quantifies the reason: PPO's "ratio deviation" (how far the policy strays from the trust region) exploded to uncontrollable values (e.g., '3689.957' in Humanoid-v4).
• SPO's performance was stable and often improved with the deeper 7-layer network. Table 1 confirms that SPO successfully kept the ratio deviation within a small, controlled bound (e.g., '0.191' in Humanoid-v4).

Environment	Index	3 layers		7 layers	
		PPO	SPO	PPO	SPO
Ant-v4	Average return (↑)	**5323.2**	4911.3	1002.8	**4672.5**
	Ratio deviation (↓)	0.229	**0.101**	548.060	**0.190**
HalfCheetah-v4	Average return (↑)	**4550.2**	3602.4	2242.3	**5307.3**
	Ratio deviation (↓)	0.225	**0.086**	1675.340	**0.188**
Hopper-v4	Average return (↑)	1119.4	**1480.3**	975.9	**1507.6**
	Ratio deviation (↓)	0.164	**0.067**	113.178	**0.194**
Humanoid-v4	Average return (↑)	795.1	**2870.0**	614.1	**4769.9**
	Ratio deviation (↓)	3689.957	**0.179**	2411.845	**0.191**
HumanoidStandup-v4	Average return (↑)	143908.8	**152378.7**	92849.7	**176928.9**
	Ratio deviation (↓)	2547.499	**0.182**	4018.718	**0.187**
Walker2d-v4	Average return (↑)	**3352.3**	2870.2	1110.9	**3008.1**
	Ratio deviation (↓)	0.170	**0.070**	998.101	**0.157**

Experiment 2: Scaling Policy Networks



A similar outcome occurred in Atari. **Figure 6** shows that when using the large ResNet-18 encoder, SPO's performance was "significantly improved," while PPO struggled to maintain control over the probability ratio, failing to leverage the more powerful network.

Experiment 3: Objective Function Analysis

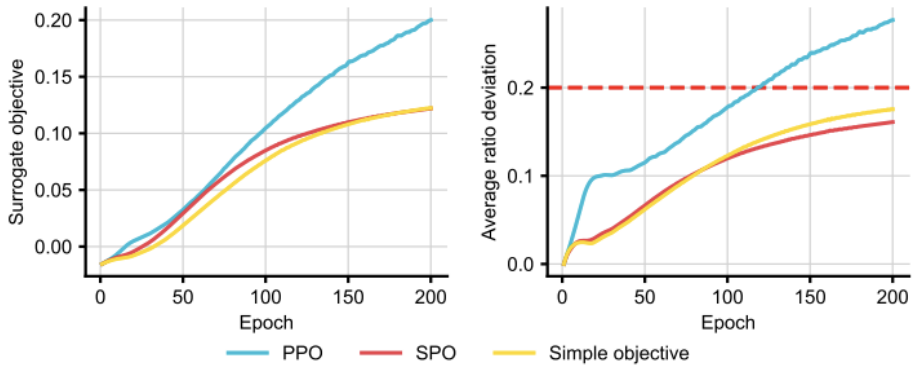
Goal: To verify that SPO's specific objective function design is superior to other possible "ϵ-aligned" objectives.

Setup: The authors compared the optimization behavior of three objectives on the same batch of data:

- PPO's clipping function (f_{ppo}),
- SPO's quadratic penalty (f_{spo}),
- and a simpler quadratic objective (f_{simple}) that is also ϵ-aligned.

Results: Figure 7 provides a clear ablation.

- PPO achieved the highest "surrogate objective" value but at the cost of an uncontrollable ratio deviation that quickly violated the $\epsilon = 0.2$ bound.
- Both f_{spo} and f_{simple} successfully constrained the ratio deviation.
- However, f_{spo} achieved a much better surrogate objective value than f_{simple}.



TEXAS
The University of Texas at Austin

CONCLUSION

Conclusion

- The paper’s experiments provide strong, multifaceted evidence supporting its claims.
- They demonstrate that SPO is a high-performing algorithm on standard benchmarks.
- More importantly, it overcomes a fundamental stability weakness in PPO, making it a more suitable and robust choice for training the large-scale neural networks common in modern deep reinforcement learning.

TEXAS
The University of Texas at Austin

THANK YOU