# Augmented World Models Facilitate Zero-Shot Dynamics Generalization From a Single Offline Environment

Philip Ball*[1], Cong Lu*[1], Jack Parker-Holder[1], Stephen J. Roberts[1]

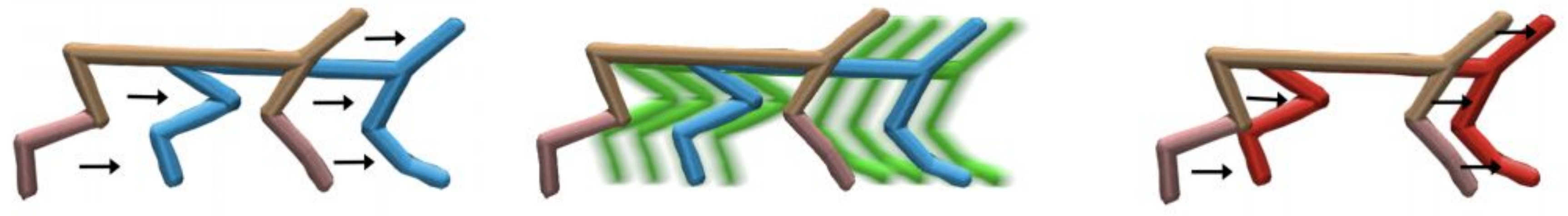
## Key Question

*How can we **generalise to novel environments** from offline data on a single environment?*

## Summary

We propose:

- Dynamics augmentation for offline RL, allowing us to be robust to changing dynamics **training only on a single setting**.
- A simple **self-supervised context adaptation** algorithm, significantly increasing zero-shot performance.
- Both approaches offer **significant improvement** v.s. SotA methods.



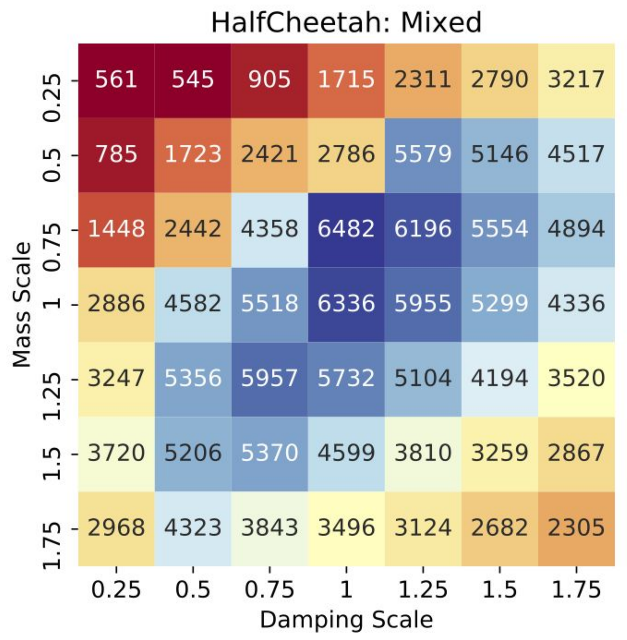
(a) World Model: $\widehat{P}$ (b) Augmented World Model: $\widehat{P}_z$ (c) Test: $P^\star$

## What happens when the dynamics change?

We train a MOPO agent on the Half-Cheetah D4RL mixed dataset.

At test time we select the mass and damping multipliers from: {0.25, 0.50, … , 1.75}.

We test the **zero-shot** performance.

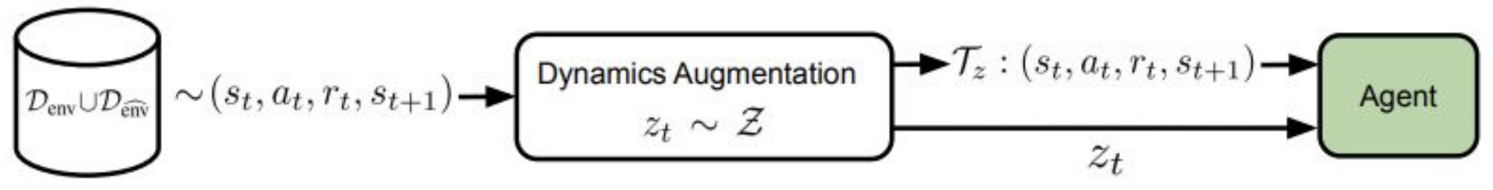



The performance can massively decrease from the default, and in many cases, the policy fails to make much progress at all. Existing offline MBRL methods cannot generalize to changed dynamics!

## Practical Algorithm: AugWM

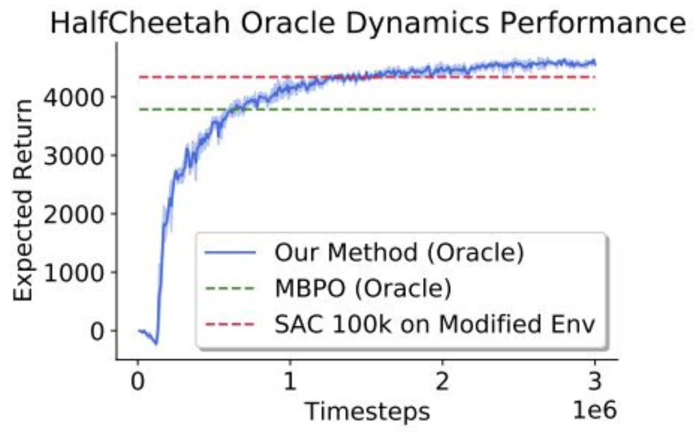
**Algorithm 1:** Augmented World Models: Training

**Input:** Offline data $\mathcal{D}_{\text{env}}$, Penalty $\lambda$, horizon $h$, batchsize $B$, augmentation $\mathcal{Z}$.
**Initialize:** Ensemble of $N$ dynamics models $\widehat{P}$, policy $\pi$. Replay buffer $\mathcal{D}_{\widehat{\text{env}}}$
1. Train $\widehat{P}$ in a supervised fashion using $\mathcal{D}_{\text{env}}$.
**for** $epoch = 1, 2, \ldots$ **do**
  (i) Sample initial states: $\{s_1^1, \ldots, s_1^B\} \sim \mathcal{D}_{\text{env}}$
  (ii) Rollout policy (in parallel), using $\lambda$-penalized reward, storing all data in $\mathcal{D}_{\widehat{\text{env}}}$
  (iii) Train policy using $\mathcal{D}_{\text{env}} \cup \mathcal{D}_{\widehat{\text{env}}}$. For each $(s, a, r, s')$, sample $z \sim \mathcal{Z}$ and apply $\mathcal{T}_z$.
**end**
**Return: Policy** $\pi$




We tested our approach with the **oracle** augmentation, i.e., the true difference between the train and test environments.

Just augmenting dynamics is sufficient for adaption!




## Sampling augmentations during training

How exactly do we approximate test-time dynamics? We consider three approaches with randomly sampled vectors $z$:

- Random Amplitude Scaling, **RAD** (Laskin et al., 2020):

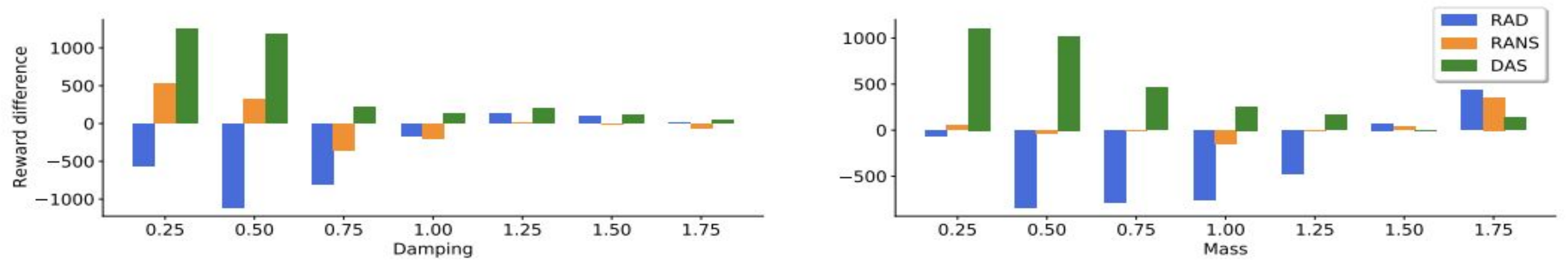
$$\mathcal{T}_z : (s_t, a_t, r_t, s_{t+1}) \mapsto (z \odot s_t, a_t, r_t, z \odot s_{t+1})$$

- Random Amplitude Nextstate Scaling, **RANS**:

$$\mathcal{T}_z : (s_t, a_t, r_t, s_{t+1}) \mapsto (s_t, a_t, r_t, z \odot s_{t+1})$$

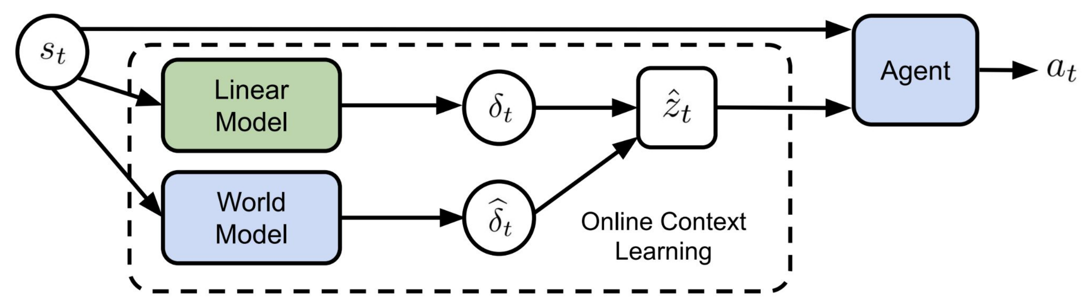
- Dynamics Amplitude Scaling, **DAS**:

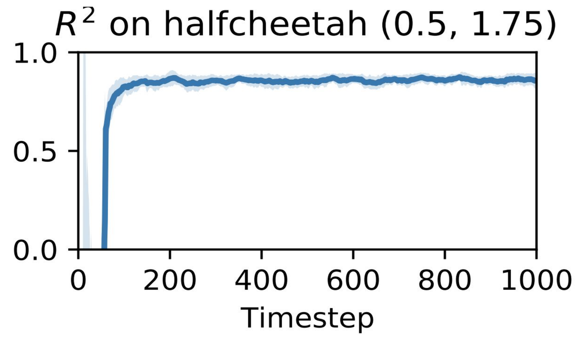
$$\mathcal{T}_z : (s_t, a_t, r_t, s_{t+1}) \mapsto (s_t, a_t, r_t, s_t + z \odot \delta_t)$$




DAS performs well even without adding the context to the policy! Simply adding this augmentation (without passing the context) makes the MOPO agent significantly more robust to changed dynamics.

## Learning the augmentation online



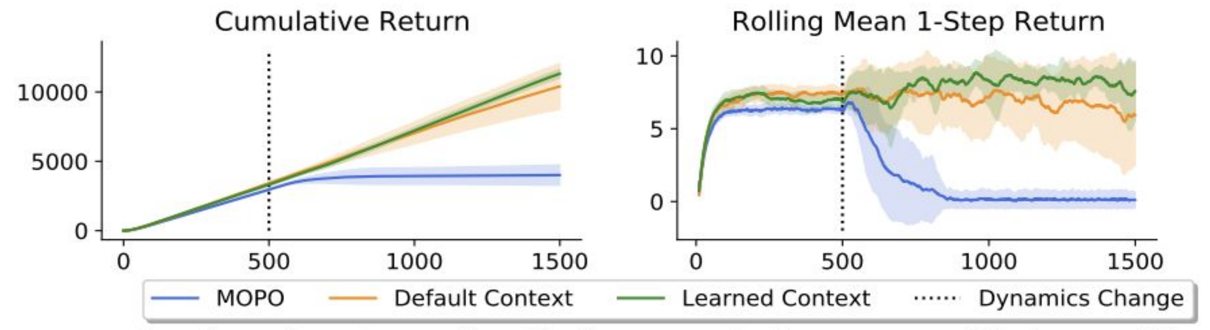

- We may Infer the augmentation at test time and then pass that to the policy as a task-descriptor allowing the policy to further adapt.
- A simple linear model trained online at test time predicting change in state from the current state produces a useful signal for predicting the correct augmentation vector $z$. An example $R^2$ of the linear model is shown above.

## Results

- The first table shows averaged results for MuJoCo changed mass/damping settings with statistically significant improvements highlighted.
- The second table shows that the DAS augmentation is suitable for more complex modified dynamics such as crippled legs and modified limb sizes.

| Dataset Type | Environment | MOPO | AugWM (Ours) |
|---|---|---|---|
| Random | HalfCheetah | 2303 ± 112 | 2818 ± 197 ⋆ |
| Random | Walker2d | 569 ± 103 | 706 ± 139 |
| Mixed | HalfCheetah | 3447 ± 218 | 3948 ± 122 ⋆ |
| Mixed | Walker2d | 946 ± 95 | 1317 ± 206 ⋆ |
| Medium | HalfCheetah | 2954 ± 89 | 2967 ± 106 |
| Medium | Walker2d | 1477 ± 337 | 1614 ± 440 |
| Med-Expert | HalfCheetah | 1590 ± 766 | 2885 ± 432 ⋆ |
| Med-Expert | Walker2d | 1062 ± 334 | 2521 ± 316 ⋆ |

| Setting | MOPO | AugWM (Default) | AugWM (LM) |
|---|---|---|---|
| Ant: Mass/Damp | 1634 | 1715 | **1804** |
| Ant: One Crippled Leg | 1370 | 1572 | **1680** |
| Ant: Two Crippled Legs | 700 | 697 | **795** |
| HalfCheetah: Big | 4891 | **5194** | 4968 |
| HalfCheetah: Small | 5151 | **5488** | 5263 |




AugWM agents can also adapt to changes in dynamics (halfcheetah-mixed, modified mass/damping) mid-episode!

## Future Work

- Meta-learning for few-shot learning: allowing the policy to also change at test time.
- Unlimited dynamics changes during test-time.
- Augmentations in latent space for pixel-based tasks.

*Equal contribution. [1]University of Oxford. Corresponding authors: ball@robots.ox.ac.uk, cong.lu@stats.ox.ac.uk