utilityfog / EZ_Optimization

Code Issues Pull requests Actions Projects Wiki Security Insights

Evaluating The Epstein-Zin Function as an ICM-PPO Optimization Target

0 stars 0 forks 0 watching Branches Activity Tags

Private repository

1 Branch 0 Tags Go to file t Go to file Add file Code

utilityfog updated requirements.txt 302112c · 8 hours ago

data	Evidence of Ungrokking?	10 hours ago
icml2026	updated requirements.txt	8 hours ago
images	Evidence of Ungrokking?	10 hours ago
src	Evidence of Ungrokking?	10 hours ago
.DS_Store	updated requirements.txt	8 hours ago
.gitignore	Initial commit	3 months ago
README.md	Completely fixed the recipe, I think.	3 days ago
icml2026.zip	updated requirements.txt	8 hours ago
requirements.txt	updated requirements.txt	8 hours ago

README

SPEC: RL INVESTOR (PPO + ICM) with Epstein–Zin Preferences + Learnable Fractional Differencing

A self-contained, implementation-ready specification. No code included. All math, shapes, distributions, rewards, targets, and losses are explicit.

---

0) PURPOSE & SCOPE

This document **replaces CRRA** with **Epstein–Zin (EZ)** recursive preferences and **adds a Learnable Fractional Differencing (FracDiff) layer** in the feature pipeline, while preserving the **PPO + ICM** training stack and environment mechanics. It is **drop-in**: policy parameterization, buffers, rollout loop, exact log-probabilities, and PPO machinery remain intact. Only the **preference aggregator** (reward/value semantics) and **feature memory module** (FracDiff) change.

What stays the same (do not touch):

- Time/indexing, assets, returns, risk-free, ~~turnover/transaction cost~~, budget identity.
- Actor heads (consumption squashed Gaussian; ~~risky weights Dirichlet/softmax~~), exact log-prob math. (No more portfolio optimization)
- Critic backbone mechanics (but we output two EZ heads; see §5).
- ICM encoder/forward (and optional inverse) and curiosity reward wiring.
- PPO: ratio, clipping, GAE, epochs/minibatching, entropy, optimizers.
- Data split, standardization, rollout collection, buffer contents.

What changes:

1. **Utility/Value:** CRRA is replaced by **Epstein–Zin** with a numerically stable target in (z)-space (§4–§6).
2. **Features:** Insert **Learnable FracDiff** over returns before feature construction (§3).

---

1) Core definitions (time, single risky asset, wealth, consumption)

We now consider a **single risky asset** (S&P) and remove portfolio optimization entirely.

1.1 Time and assets

- Discrete time ($t = 0, 1, 2, \ldots, T-1$).
- One risky asset with **gross** return ($R_{t+1} \in R_{>0}$) between (t) and ($t+1$).
- No explicit risk–free asset and no portfolio weights – all *unconsumed* wealth is automatically invested in the risky asset.

1.2 Wealth, consumption, normalization

- Wealth at start of step (t): ($W_t > 0$).
- Consumption fraction (action): ($c_t \in (0,1)$); **dollar consumption** ($C_t := c_t \cdot W_t$).
- Running max wealth ($M_t := \max_{0 \le \tau \le t} W_\tau$); normalized wealth ($\tilde{W}_t := W_t / M_t \in (0,1]$).

1.3 Budget identity (wealth transition)

In the simplified world, after consuming ($C_t = c_t W_t$), the remaining wealth ($(1-c_t)W_t$) is fully invested in the risky asset, which realizes a gross return (R_{t+1}) over ($[t, t+1]$).

The **wealth transition** is

$$([W_{t+1} = (1-c_t)W_tR_{t+1}\,.])$$

We may optionally clip (W_{t+1}) below by a small floor $(\varepsilon_W > 0)$ for numerical stability. Running max wealth is updated as

$$([M_{t+1} := \max(M_t, W_{t+1})\,.])$$

---

2) OBSERVATIONS, FEATURES, STATE (BASELINE PIPELINE)

2.1 Observables at time (t)

- (W_t) and a causal feature vector $(x_t \in R^d)$ built **only** from data $(\le t)$.
- Standardize (x_t) via train-set $((\mu, \sigma))$ to $(\tilde{x}_t)$ (store (μ, σ) from training only).

2.2 State to networks

- **State:** $(s_t := \operatorname{concat}\left(\tilde{W}_t, \tilde{x}_t\right) \in R^{1+d+n})$ (fixed order).

At time (t) the agent observes:

- Normalized wealth $(\tilde{W}_t = W_t \,/\, M_t)$.
- A standardized feature vector $(\tilde{x}_t \in R^d)$, built from the FracDiff pipeline and other signals, using only information up to time (t).

The **state** fed to the policy and critic is

$$(s_t := \operatorname{concat}\left(\tilde{W}_t, \tilde{x}_t\right) \in R^{1+d}\,.)$$

There is no (w_{t-1}) term in the state any more, since there is no portfolio decision.

---

3) Learnable Fractional Differencing (returns-domain feature module)

3.1 Goal & parameter

Learn a memory depth $(d_{\text{target}} \in [d_{\min}, d_{\max}])$ (e.g., $([0,1])$) that controls the fractional differencing of returns to **capture long memory** while promoting **stationarity**.

3.2 Placement in pipeline

- Input raw **log-returns** per asset: $(r_t \in R^n)$ (or windows).
- Apply a FracDiff operator with effective exponent (d_{eff}):
  - **Mode "direct"**: apply $((1-L)^{d_{\text{target}}})$ to returns.
  - **Mode "price_equiv"**: apply $((1-L)^{d_{\text{target}}-1})$ to returns (equivalent to price fracdiff of (d_{target}) without reconstructing prices).

- Truncate the kernel to length (K) (auto-chosen from (d_{eff}) and a tolerance). Outputs lose the first (K) steps.

3.3 State augmentation & alignment

- Build usual statistics **from** the FD output (lags, MAs, vol, PCA, cross-sectional transforms).
- **Shift** all time-aligned targets by (K) (drop first (K) steps) so shapes match.
- Optionally append $(\text{stop}_{\text{grad}}(d_{\text{target}}))$ and (K) as scalar features so the policy/critic can adapt to memory depth.

3.4 Regularization & constraints

- Keep (d_{target}) within bounds via a sigmoid reparameterization.
- Add a small L2 penalty if (d_{target}) sticks to the bounds.
- Optional "whiteness" regularizer: penalize low-lag autocorrelation of FD residuals to avoid over-memory.

> **Everything backpropagates end-to-end** because kernel weights are differentiable functions of (d_{eff}).

---

4) Epstein–Zin Preferences (replace CRRA)

Let $(\beta \in (0,1))$ be the subjective discount, $(\gamma > 0)$ risk aversion, $(\psi > 0)$ intertemporal elasticity (EIS). Define transforms:

- $(z(V) := V^{1-\frac{1}{\psi}})$ (EIS/consumption space)
- $(y(V) := V^{1-\gamma})$ (risk space)

4.1 EZ aggregator (Kreps–Porteus form)

For lifetime utility (V_t) and consumption (C_t):

$$([V_t = \left[(1-\beta)C_t^{1-\frac{1}{\psi}} + \beta\left(E_t[V_{t+1}^{1-\gamma}]\right)^{\frac{1-\frac{1}{\psi}}{1-\gamma}} \right]^{\frac{1}{1-\frac{1}{\psi}}}.])$$

4.2 Practical RL parameterization (stable targets)

We **train in** (z)**-space** with a two-head critic predicting $(\hat{z}_t \approx z(V_t))$ and $(\hat{y}_t \approx y(V_t))$.

- **External (shaped) reward:** $(r_t^{\text{ext}} := (1-\beta)C_t^{1-\frac{1}{\psi}})$.
- **One-step bootstrap target for** (z)**:**

$$([T_t^{(z)} := (1-\beta)C_t^{1-\frac{1}{\psi}} + \beta\left(\hat{y}_{t+1}\right)^{\frac{1-\frac{1}{\psi}}{1-\gamma}}.])$$

- **Value loss:** $(L_{\text{value}} := \frac{1}{2}\left(\hat{z}_t - T_t^{(z)}\right)^2)$.

- Optional **consistency** regularizer: encourage ($\hat{y}_t \approx (\hat{z}_t)^{\frac{1-\gamma}{1-\frac{1}{\psi}}}$) with a small weight.

> **Degeneracies:** ($\psi \to 1$) approaches additive/separable (log-like); ($\gamma \to 1$) reduces risk curvature; recipe reduces toward CRRA smoothly.

---

5) ACTOR & CRITIC (Z-functions, distributions, exact log-probs)

We now have a **single action dimension**: the consumption rate ($c_t \in (0,1)$).

Dimensions and symbols used throughout this section

- State at time (t): ($s_t \in R^{1+d+n}$) is the concatenation ($s_t := \text{concat}\left(\tilde{W}_t, \tilde{x}_t\right)$), where ($\tilde{W}_t = W_t \ / \ M_t$) and ($\tilde{x}_t$) is the standardized feature vector.
- Consumption fraction (action component): ($c_t \in (0,1)$). Dollar consumption: ($C_t := c_t W_t$).
- Hyperparameters for heads: ($\sigma_{\min} > 0$) (std floor).

5.1 Actor (f_θ)

The actor takes ($s_t \in R^{1+d}$) and passes it through a shared backbone (e.g. an MLP) to produce parameters for a scalar Gaussian in a latent space:

- Pre–squash Normal parameters: ($[\mu_c(s_t) \in R, \quad \ell_c(s_t) \in R, \quad \sigma_c(s_t) := \text{softplus}(\ell_c) + \sigma_{\min} .]$)
- Sample pre–squash variable ($[y_c \sim N\left(\mu_c(s_t), \sigma_c(s_t)^2\right) .]$)
- Squash to the action space via the sigmoid: ($[c_t := \sigma(y_c) = \frac{1}{1+e^{-y_c}} \in (0,1) .]$)
- Deterministic (evaluation) action is given by ($[c_t^{\text{det}} := \sigma\left(\mu_c(s_t)\right) .]$)

There is no risky–weights head any more; (w_t) is implicitly equal to (1) on the single asset.

5.2 Exact log–probability of (c_t)

Let

$$([y_c = \text{logit}(c_t) = \log\frac{c_t}{1-c_t} .])$$

The log–probability under the squashed Gaussian is

$$([\log p(c_t \mid s_t) = \log N\left(y_c, \mu_c(s_t), \sigma_c(s_t)^2\right) - \log\left(c_t(1-c_t)\right),])$$

where the first term is the Gaussian log–density of (y_c) and the second term is the log–Jacobian of the sigmoid.

This ($\log p(c_t \mid s_t)$) is the **only** action log–probability used in PPO here.

5.3 Critic (g_ψ) (two heads for EZ)

The critic takes (s_t) and outputs two scalars:

- $(\hat{z}_t \approx z(V_t))$ with $(z(V) := V^{1-\frac{1}{\psi}})$.
- $(\hat{y}_t \approx y(V_t))$ with $(y(V) := V^{1-\gamma})$. These are used to build the EZ bootstrap target and TD residual below.

---

6) ENVIRONMENT STEP (FULL SEQUENCE)

At time (t), given state (s_t) and sampled consumption rate (c_t), the environment performs:

1. **Consumption and wealth evolution**

Dollar consumption:

$$([C_t = c_t W_t\,.])$$

Remaining wealth:

$$([W_t^{\text{after}} = (1 - c_t) W_t\,.])$$

Apply the risky asset gross return (R_{t+1}):

$$([W_{t+1} = W_t^{\text{after}} R_{t+1} = (1 - c_t) W_t R_{t+1}\,.])$$

Optionally clip $(W_{t+1} \geq \varepsilon_W)$ if needed for numerical stability.

2. **Running max and next state**

$$([M_{t+1} = \max(M_t, W_{t+1}), \quad \tilde{W}_{t+1} = \frac{W_{t+1}}{M_{t+1}}\,.])$$

The feature pipeline (including FracDiff) produces the next standardized feature vector $(\tilde{x}_{t+1})$ from market data up to time $(t+1)$.

The next state is

$$([s_{t+1} = \operatorname{concat}\left(\tilde{W}_{t+1}, \tilde{x}_{t+1}\right)\,.])$$

3. **Termination**

   Episodes end when $(t = T - 1)$ (or when data runs out).

Wealth transition

- Gross growth factor:

$$([G_{t+1} := (1 - c_t)\left(R_f[t+1] + w_t^\top \tilde{R}[t+1]\right) - \kappa \|w_t - w_{t-1}\|_1\,.])$$

- Next wealth: ($W_{t+1} := W_t \cdot G_{t+1}$). Safety floor may clip ($G_{t+1} \geq \varepsilon_g > 0$).

7) REWARDS (EXTERNAL EZ FLOW, INTRINSIC ICM)

We use the Epstein–Zin flow term for external reward and the Intrinsic Curiosity Module (ICM) to supply an intrinsic shaping signal.

EZ parameters

- Discount ($\beta \in (0,1)$)
- Risk aversion ($\gamma > 0$)
- Elasticity of intertemporal substitution (EIS) ($\psi > 0$)
- Consumption ($C_t = c_t W_t$)

7.1 External reward (EZ flow term in (z)-space)

The **external reward** at time (t) is the EZ flow term in (z)–space, depending only on consumption:

$$([r_t^{\text{ext}} = (1-\beta)C_t^{1-\frac{1}{\psi}} = (1-\beta)(c_t W_t)^{1-\frac{1}{\psi}}.])$$

This is the main objective that encourages good consumption timing.

7.2 Intrinsic Curiosity Module (ICM) — complete specification

We define the ICM exactly and fully:

Network dimensions

Let:

- Feature dimension (d)
- State dimension ($1+d$) since state is ($\mathrm{concat}(\tilde{W}_t, \tilde{x}_t)$)
- State-embedding dimension (m) (e.g., 64)
- Hidden widths for ICM networks:
  - Encoder hidden width (E) (e.g., 128)
  - Forward-model hidden width (F) (e.g., 128)

Define the state encoder:

$$([\phi_\omega : R^{1+d} \to R^m.])$$

State encoder network

Given state ($s_t \in R^{1+d}$):

$$([e1 = \mathrm{GELU}(W_{e1}s_t + b_{e1}), \quad W_{e1} \in R^{E\times(1+d)}.])$$

$$([e2 = \mathrm{GELU}(W_{e2}e1 + b_{e2}), \quad W_{e2} \in R^{E\times E}.])$$

$$([\phi(s_t) = W_{eo}e2 + b_{eo}, \quad W_{eo} \in R^{m\times E}.])$$

Define:

$$([\phi_t := \phi(s_t), \qquad \phi_{t+1} := \phi(s_{t+1}).])$$

Action embedding (scalar action)

Because the action is **only** consumption ($c_t \in (0,1)$), we embed it as:

$$([y_c = \mathrm{logit}(c_t) = \log\frac{c_t}{1-c_t}.])$$

Then define:

$$([\psi(a_t) := \psi(c_t) := y_c \in R^1.])$$

Dimensions: action embedding is 1-dimensional.

Forward dynamics model

Maps $((\phi(s_t), \psi(a_t)))$ into a prediction of $(\phi(s_{t+1}))$.

Input dimension to forward model:

$$([m + 1.])$$

Forward model layers:

$$([u1 = \mathrm{GELU}\left(W_{f1}, \mathrm{concat}(\phi_t, \psi(c_t)) + b_{f1}\right), \quad W_{f1} \in R^{F\times(m+1)}.])$$

$$([u2 = \mathrm{GELU}(W_{f2}u1 + b_{f2}), \quad W_{f2} \in R^{F\times F}.])$$

$$([\hat{\phi}_{t+1} := W_{fo}u2 + b_{fo}, \quad W_{fo} \in R^{m\times F}.])$$

There is **no inverse model** in the consumption-only version.

Intrinsic reward

Given:

- Encoded next state (ϕ_{t+1})
- Predicted next state $(\hat{\phi}_{t+1})$

The intrinsic reward is:

$$([r_t^{\text{int}} := \eta \left|\phi_{t+1} - \hat{\phi}_{t+1}\right|_2^2,])$$

with a small scale factor $(\eta > 0)$ (e.g., (10^{-3})).

ICM losses

Forward loss:

$$([L_{\text{fwd}}(\omega) := \left|\phi_{t+1} - \hat{\phi}_{t+1}\right|_2^2 .])$$

Inverse loss:

$$([L_{\text{inv}} := 0])$$

since we removed portfolio weights and do not reconstruct (w_t).
The action is 1-dimensional and directly known, so inverse dynamics is unnecessary.

Total ICM loss:

$$([L_{\text{ICM}} = L_{\text{fwd}} .])$$

7.3 Total reward used by PPO

$(r_t := r_t^{\text{ext}} + r_t^{\text{int}})$.

---

8) Advantages, EZ targets, and losses (consumption-only)

All variables used below are defined here or earlier sections.

8.1 EZ one-step target in (z)-space

We have two critic heads:

- $(\hat{z}_t \approx z(V_t) := V_t^{1-\frac{1}{\psi}})$
- $(\hat{y}_t \approx y(V_t) := V_t^{1-\gamma})$

The one-step EZ bootstrap target is:

$$([T_t^{(z)} = (1-\beta)C_t^{1-\frac{1}{\psi}} + \beta\left(\hat{y}_{t+1}\right)^{\frac{1-\frac{1}{\psi}}{1-\gamma}} .])$$

All terms are fully defined:

- ($C_t = c_t W_t$) is consumption
- ($\hat{y}_{t+1}$) comes from the critic applied to next state
- exponents come from EZ preference structure

8.2 Value loss (z-head)

$$([L_{\text{value}} = \frac{1}{2}\left(\hat{z}_t - T_t^{(z)}\right)^2 .])$$

8.3 TD residual in (z)-space and GAE

Define the **combined reward**:

$$([r_t = r_t^{\text{ext}} + r_t^{\text{int}}])$$

and the **EZ temporal-difference residual**:

$$([\delta_t^{\text{EZ}} := r_t + \beta\left(T_t^{(z)} - r_t^{\text{ext}}\right) - \hat{z}_t .])$$

This matches the structure of the general EZ TD residual while incorporating intrinsic reward.

Generalized Advantage Estimation (GAE)

Let ($\lambda \in [0, 1]$) be the GAE parameter.
Compute the advantages by backward recursion:

$$([\tilde{A}_t = \delta_t^{\text{EZ}} + (\beta\lambda), \delta_{t+1}^{\text{EZ}} + (\beta\lambda)^2, \delta_{t+2}^{\text{EZ}} + \cdots])$$

Practical implementation uses backward iteration over a rollout.
We normalize ($\tilde{A}_t$) to mean 0 and variance 1 in each minibatch.

8.4 PPO clipped policy loss (consumption-only)

Let:

- ($\log \pi_{\theta_{\text{old}}}(c_t \mid s_t)$) be the stored behavior log-prob.
- ($\log \pi_{\theta}(c_t \mid s_t)$) be recomputed with the current actor.
- Importance ratio:

$$([r_t(\theta) := \exp\left(\log \pi_{\theta}(c_t \mid s_t) - \log \pi_{\theta_{\text{old}}}(c_t \mid s_t)\right) .])$$

- Clipping parameter ($\varepsilon \in (0, 1)$).

The PPO objective (to **minimize**) is:

$$([L_{\text{PPO}} = -E_t\Big[\ \min\left(r_t(\theta)\tilde{A}_t, \text{clip}(r_t(\theta), 1-\varepsilon, 1+\varepsilon)\tilde{A}_t\right)\Big] .])$$

8.5 Entropy term (Gaussian action only)

The actor samples

$$([y_c \sim N(\mu_c(s_t), \sigma_c(s_t)^2)])$$

before applying the sigmoid.

Entropy of a Normal:

$$([H_c = \frac{1}{2}\log\left(\, 2\pi e \sigma_c^2 \,\right) .])$$

We encourage exploration by adding the entropy term:

$$([L_{\text{ent}} := \ -H_c\,.])$$

There is no Dirichlet entropy here since we removed risky-weight allocations.

8.6 ICM loss

As defined in §7.2:

$$([L_{\text{ICM}} = L_{\text{fwd}} = \left| \phi_{t+1} - \hat{\phi}_{t+1} \right|_2^2 .])$$

The inverse loss is zero in consumption-only and can be enabled later if desired.

8.7 Final training loss

Define scalar weighting hyperparameters:

- $(c_v > 0)$: value loss weight
- $(\beta_{\text{ent}} > 0)$: entropy loss weight
- $(c_{\text{icm}} > 0)$: curiosity loss weight

The full objective is:

$$([L_{\text{total}} = L_{\text{PPO}} + c_v L_{\text{value}} + \beta_{\text{ent}} L_{\text{ent}} + c_{\text{icm}} L_{\text{ICM}}\,.])$$

---

9) TRAINING PROCEDURE (COLLECT → TARGETS → PPO)

9.1 Hyperparameters (additions/changes)

- **EZ:** choose $(\gamma \in \ 5, 10)$, $(\psi \in \ 0.5, 1.0, 1.5)$, $(\beta \in [0.95, 0.999])$.
- **FracDiff:** (d_{target}) init (0.3) – (0.5) within $([0, 1])$, tolerance (10^{-4}), $(K_{\max} \in [1024, 4096])$ (match horizon).

- **RL:** keep PPO (λ), clip, epochs, minibatch, lrs same initially.

9.2 Rollout collection (unchanged mechanics)

- Collect tuples ($\left(s_t, a_t = (c_t, w_t), r_t, s_{t+1}, \log\pi_{\theta_{\text{old}}} \right)$) where (r_t) includes EZ flow + curiosity.
- Align time by dropping first (K) steps due to FracDiff.

9.3 Target building & PPO update

- For each step, compute ($T_t^{(z)}$), (δ_t^{EZ}), GAE, and (z)-value loss.
- Recompute current ($\log\pi_\theta$) exactly (§5.2); perform clipped PPO with entropy and ICM losses.
- After epochs, set ($\theta_{\text{old}} \leftarrow \theta$).

9.4 Evaluation (deterministic)

- Use ($c_t := \sigma(\mu_c)$), ($w_t := \alpha \ / \ \sum_i \alpha_i$).
- Recover EZ value via inverse transform for reporting: ($\hat{V}_t = \hat{z}_t^{\,1/(1-\frac{1}{\psi})}$).
- Report PnL, CAGR, MDD, Calmar, turnover, and ($\hat{V}_0$).

---

10) DIAGNOSTICS, CHECKS, & ABALATIONS

- **EZ sanity:** as ($\psi \to 1$) or ($\gamma \to 1$), curves and training behavior should smoothly approach separable/CRRA.
- **Scale hygiene:** track ($\hat{z}_t, \hat{y}_t$) magnitudes; clamp/normalize if exploding.
- **FracDiff:** monitor learned (d_{target}) trajectory; inspect ACF/PACF of FD outputs; avoid non-stationary drift.
- **Alignment:** verify all post-FD tensors drop the first (K) steps; shapes of policy/value/ICM batches match.
- **Ablations:** (i) turn off FracDiff (identity) to test EZ alone; (ii) (ψ) grid with fixed (γ); (iii) compare CRRA vs EZ at matched (γ) with ($\psi \approx 1$).

---

11) MINIMAL MIGRATION CHECKLIST

- [ ] Expose (γ, ψ, β) in config; leave PPO hypers unchanged initially.
- [ ] Critic: switch to **two heads** (($\hat{z}$, $\hat{y}$)); keep shared backbone.
- [ ] Reward pipe: compute ($r_t^{\text{ext}} = (1-\beta)C^{1-1/\psi}$); add curiosity as before → (r_t).
- [ ] Targets: build ($T^{(z)}$) with next-state ($\hat{y}$); compute (δ^{EZ}) and GAE in (z)-space.
- [ ] Insert **Learnable FracDiff** before feature builder; shift by (K).
- [ ] Log (d_{target}, K), ACF diagnostics, and ($\hat{z}$, $\hat{y}$) summaries.

---

12) GLOSSARY

- (C_t): dollar consumption; (W_t): wealth; (c_t): consumption rate.
- (γ): risk aversion; (ψ): EIS; (β): discount.
- (V_t): EZ lifetime utility; $(z(V) = V^{1-1/\psi})$; $(y(V) = V^{1-\gamma})$.
- (d_{target}): fracdiff memory parameter; (K): kernel truncation length.
- ICM: Intrinsic Curiosity Module; (ϕ): encoder; (f): forward model.

---

13) TL;DR (one-screen summary)

- **Objective change:** CRRA → **Epstein–Zin** with a two-head critic $((\hat{z}, \hat{y}))$, shaped external reward $((1-\beta)C^{1-1/\psi})$, and a one-step (z)-target using $(\hat{y}_{t+1})$.
- **Feature change:** Insert **Learnable FracDiff** over returns; align time by kernel length (K).

Releases

No releases published
Create a new release

Packages

No packages published
Publish your first package

Languages

- **TeX** 58.1%
- **Python** 26.2%
- **BibTeX Style** 15.7%

Suggested workflows

Based on your tech stack

Python Package using Anaconda Configure

Create and test a Python package on multiple Python versions using Anaconda for package management.

SLSA Generic generator Configure

Generate SLSA3 provenance for your existing release workflows

Python package Configure

Create and test a Python package on multiple Python versions.

More workflows

Dismiss suggestions