Probabilistic Prediction of Stochastic Dynamical Systems

A ceiling, a floor, and an open-source implementation of both

Tao Xu

Prof. Jianping He

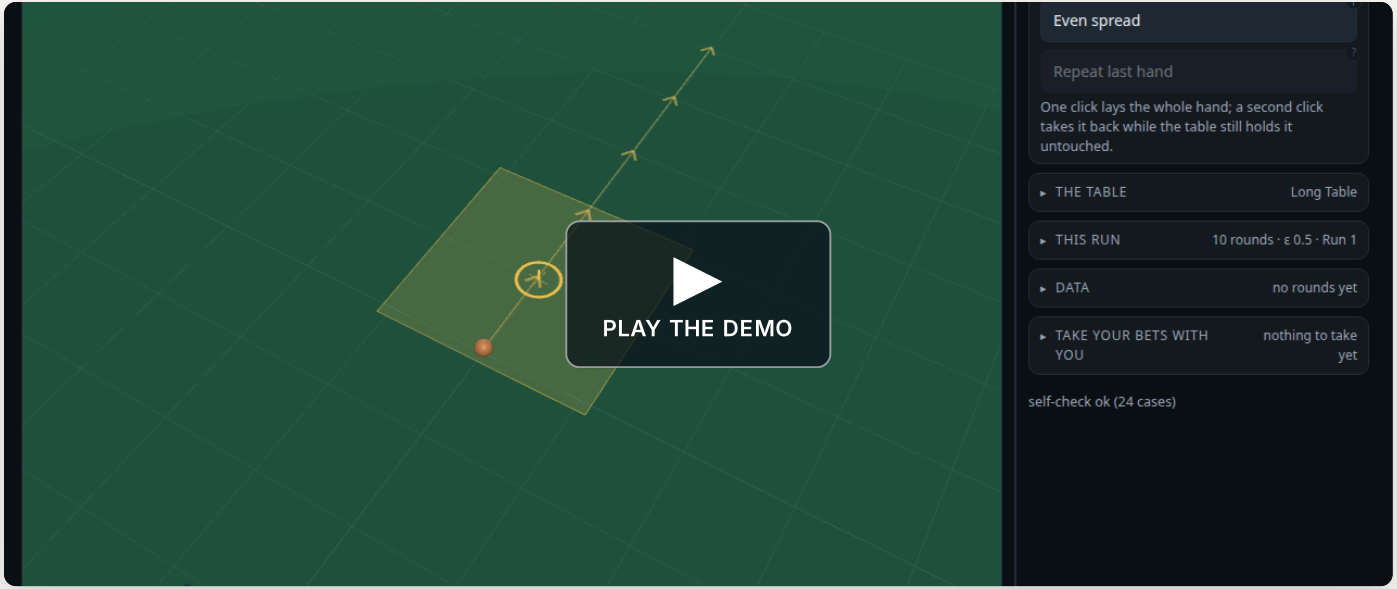
School of Automation and Intelligent Sensing

Shanghai Jiao Tong University

OPEN SOURCE `ppsds` 0.0.3-alpha · github.com/Xuzerken/ppsds

The game in one take

A minute on your phone, and you have placed a probabilistic prediction.



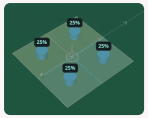
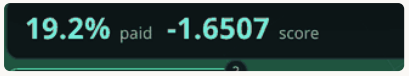
Play on your phone · www.tao-xu.com/ppsds/ · examples/understand_ppsds_by_pmf_betting/game/

Setting the problem

From the game to the question

The game in three objects

> **The game already contains the three objects of the problem — each has a name in the library and a definition in mathematics.**

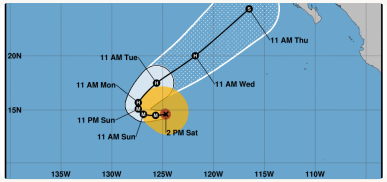
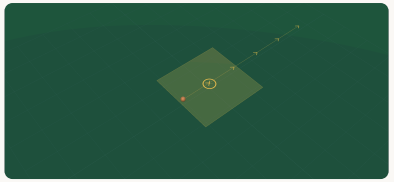
on the table	in `ppsds`	in math
 the puck	`AdditiveDriftSds(f, p_w)`	$x_{k+1} = f_k(x_k, u_k) + w_k$
 the stacks	`DriftShiftedAtomPmfPredictor( AtomPmf(...))`	$\hat{p}(\cdot \mid x_k, u_k)$
 the payout	`log(AtomPmf. box_integral(x, eps))`	$\mathcal{L}_\varepsilon(\hat{p}, x) = \log \int_{\lvert s-x\rvert_\infty \le \varepsilon} \hat{p}(s)\,\mathrm{d}s$

The first column is cut out of the game; the second is the call you would write; the third is the same thing stated formally.

> → *This pattern is not special to this betting game.*

The same pattern, everywhere

> **Different worlds, one object: a state, a known drift, and an unknown kick.**

	 **Weather / climate**	 **Robotics / a drone**	 **The game**
state	atmosphere	pose	puck
drift	physics	controller	table law
kick	unresolved scales	wind and slip	bounce
decision it informs	storm risk	safe planning	a bet

Images: NHC, public domain · A. Burden, CC0.

> → *Different names — so we need one mathematical object.*

What is a stochastic dynamical system?

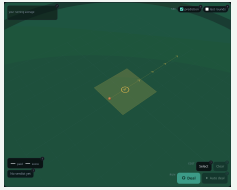
A stochastic dynamical system is the part you know plus the part you do not.

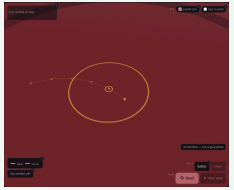
$$x_{k+1} = f_k(x_k, u_k) + w_k$$

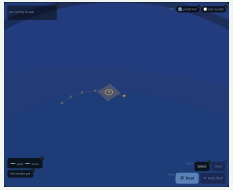
- f_k — the dynamics: the part you know, or think you know.
- u_k — the control input: the part you choose.
- w_k — the process noise: the part you do not know.

IN THE GAME

Every scenario in the betting game is like that: a nominal model, an input you apply, and a kick you did not model.


Long Table


Roulette


Whirlpool

→ *If the next state is random, what should a prediction return?*

From a point to a distribution

> **The game scores every round; to turn that into a general metric, start from a point forecast.**

IN THE GAME

One stack is a **point forecast**; covering the table is a **bet over places**. The score is the metric.

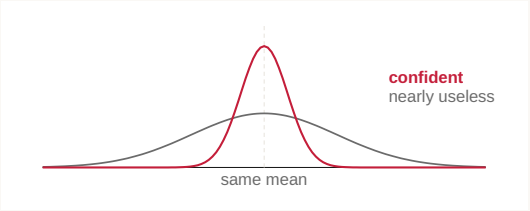


one place



a bet over places

To generalise it, go back to the point forecast — and see what it cannot say.



Same mean, different variance — a point cannot tell them apart.

A distribution is what lets you talk about **risk**, **tails** and **rare events** — the things a decision actually depends on.

> → *To compare the performance of two predicted distributions, we need a metric.*

How do we score a distribution?

The true distribution is never available — so a score is the only thing we can compute.

Point
distance $\|x - \hat{x}\|_2$

→

Distribution
a pdf distance $D_{\mathrm{KL}}(\hat{p} \,\|\, p)$

→

The obstacle
the true pdf is never available

What people use

option	what it gives	the catch
scoring rules (log, CRPS, energy)	one number; optimisable	needs a proper choice; log is tail-sensitive
moment errors (mean, covariance)	interpretable and cheap	ignores shape and tails
calibration + sharpness	a diagnostic	not a single objective

The scoring rule — the bridge that never needs the true pdf:

$$S(\hat{p}, x) : \text{prediction} + \text{outcome} \mapsto \text{a number}$$

→ *Which scoring rule is the most common?*

The logarithm score

Properness is what makes a scoring rule honest; the log score is the proper, local one.

1. **Proper.** With $S(\hat{p}, p) = \mathbb{E}_{x\sim p} S(\hat{p}, x)$, a rule is proper if $S(\hat{p}, p) \leq S(p, p)$ for every $\hat{p}$ — the truth is optimal. Strict if equality holds only at $\hat{p} = p$.

2. **The logarithm score.** $\mathcal{L}(\hat{p}, x) = \log \hat{p}(x)$ is proper and **local**; up to equivalence it is the only local proper scoring rule [Bernardo 1979; Gneiting & Raftery 2007; Parry et al. 2012]. But it reads a single point — and a single point can send it to $-\infty$.

REMARK

A second reading: differential privacy. For the **plain** log score ($\varepsilon = 0$), two neighbouring datasets are ε-indistinguishable when their score gap is bounded by ε. *A privacy budget — not the tolerance radius.* An observation from our ongoing work; collaborations welcome.



→ *So we read a neighbourhood instead.*

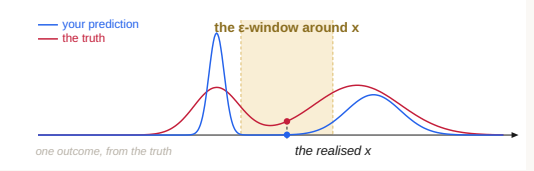
Why ε, and not 0?

> **If the model is slightly wrong, the plain score falls to $-\infty$ — because it reads a single point.**

Definition 1 *(ε-log score · Auto. 2025, §III-C)*

For a radius $\varepsilon \geq 0$ and the realized outcome x, the score reads the **probability mass in a box**, not the density at a point:

$$\mathcal{L}_\varepsilon(\hat{p}_{\mathbf{x}}, x) := \begin{cases} \log \hat{p}_{\mathbf{x}}(x), & \varepsilon = 0 \\ \log \int_{\|s-x\|_\infty \leq \varepsilon} \hat{p}_{\mathbf{x}}(s)\, \mathrm{d}s, & \varepsilon > 0 \end{cases}$$



Same prediction: read at a point, and over a window.

IN THE GAME

A stack pays if the ball lands within ε.



The box has side 2ε: what removes the $-\infty$ is reading a **neighbourhood**, not a normalisation. **The score is a bet** — the density is your stake, the box is what pays, and ε is how much risk you accept.

→ *So what is a probabilistic prediction over a whole trajectory?*

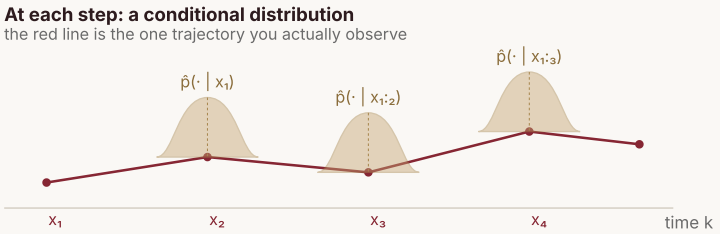
A predictor is a policy: at every step, it returns a conditional distribution.

At step k, given everything up to now:

$$\hat{p}_{\mathbf{x}_k|\mathbf{x}_{1:k-1}}\left(\cdot \mid x_{1:k-1}\right)$$

What we score is the **trajectory average**:

$$\bar{\mathcal{L}}_\varepsilon(\hat{p}_{\mathbf{x}_{1:T}}, x_{1:T}) = \frac{1}{T}\sum_{k=1}^{T} \mathcal{L}_\varepsilon\left(\hat{p}_{\mathbf{x}_k|\mathbf{x}_{1:k-1}}(\cdot \mid x_{1:k-1}), x_k\right)$$



One realized path — a distribution at every step.

- **You only ever see one trajectory** — the one the machine actually ran.
- A predictor is a **policy**: a rule from history to a distribution at every step.
- The score is the ε-log score; ε is chosen, not learned.

→ *That is a sequential decision problem — and it has an MDP shape.*

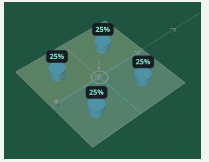
The same problem, as an MDP

The prediction problem is an MDP whose action is a distribution.

MDP	this problem
state	state-control pair $z_k = (x_k, u_k)$
action	predictive pdf $\hat{p}_k$
policy π	predictor $\mathscr{F}$
reward	log score $\mathcal{L}(\hat{p}_k, x_{k+1})$
return	cumulative score J
environment P	system kernel in $\mathcal{I}_k(z)$
value V^*	worst-case expected score
Bellman equation	the same recursion

IN THE GAME

An action is a bet, not a number — the action space is the space of pdfs; that is why this is a *functional* problem.



In MDP the question is **how good a policy can be**; it splits by what you know — and `ppsds` can ask it on both sides:

System known → optimal control

System unknown → RL

→ *So how good can a probabilistic predictor be?*

The question

How good can a probabilistic predictor be?

- **System known** (the optimal-control side): a **ceiling** any predictor can reach — and it is computable.
- **System unknown** (the RL side): a **floor** that holds over every system you are willing to admit.

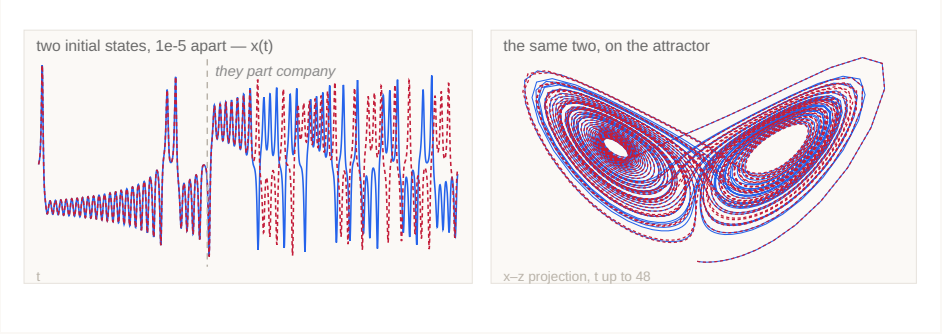
→ *Part I answers the first half.*

How predictable is a system?

Part I — the ceiling

Chaos answered the exactly-known case

> **Chaos gave a complete answer to the limit of prediction — for a system you know exactly.**



Lorenz (1963): two states 1e-5 apart track each other, then end up far apart.

- **The question it asked.** How far can you predict a system whose equations you know *exactly*?
- **What it delivered.** A complete answer — sensitive dependence, and a horizon set by the Lyapunov exponent.
- **The premise it never questioned.** *Exactly* — the one thing a real system never hands you.

> → *But real systems are not a deterministic equation you know exactly — so the question has to change.*

A line of research

> **"How well can we predict?" has been asked for decades; the limit for stochastic systems is still not understood from the probabilistic side.**



Edward N. Lorenz

raised predictability as a question

"Predictability: A problem partly solved"

1996



Andrew B. Nobel

optimality of sequential prediction (theory)

"On optimal sequential prediction for general processes"

2003



Chaoming Song

limits of specific systems (empirical)

"Limits of predictability in human mobility"

2010

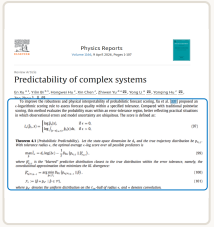


Richard Kleeman

the information-theoretic view

"Information Theory and Dynamical System Predictability"

2011



Zhou et al.

Physics Reports review (cites this work)

"Predictability of Complex Systems"

2026

→ *And this limit can, in fact, be written as a closed form.*

A predictor is a function; the ceiling is how high the expected score can go.

$$(\mathbf{P}): \quad \sup_{\hat{p}_{\mathbf{x}_k|\mathbf{x}_{1:k-1}} \in \mathcal{P},\ \forall k \in [1,T]} \ \bar{\mathcal{L}}_\varepsilon(\hat{p}_{\mathbf{x}_{1:T}}, p_{\mathbf{x}_{1:T}})$$

You choose the predictor — a **function** from history to a predictive pdf.

Nature draws the trajectory — the system p is fixed, not chosen against you.

no adversary yet — **Part II adds one.**

→ *So how high can that maximum be?*

Three results

> **This part gives three things: a ceiling, a score for any predictor, and one trajectory that is enough.**

C1 **The ceiling, in closed form.**

The best expected score any predictor can reach — determined by the tolerance, the noise entropy and the dimension, and by nothing about your method.

C2 **The score of an arbitrary predictor.**

Not only the best one: the expected score of the predictor you already have also has a closed form.

C3 **One trajectory is enough.**

You only ever see one run of the machine, and the time average still converges to the expected score.

> *The next three pages carry these numbers as C1 / C2 / C3; the fourth page puts them in the library.*

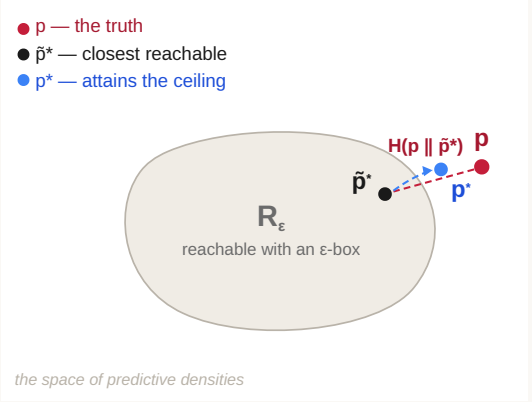
The ceiling is a closed form — and your model is not one of its arguments.

Theorem 1 *(Probabilistic predictability · Auto. 2025, §IV)*

(i) The ceiling, for $\varepsilon \geq 0$ and any $T \geq 1$ — the max is over **every step's** predictor ($\forall k$):

$$\max_{\hat{p}_{\mathbf{x}_k|\mathbf{x}_{1:k-1}} \in \mathcal{P}\ \forall k \in [1,T]} \bar{\mathcal{L}}_\varepsilon(\hat{p}_{\mathbf{x}_{1:T}}, p_{\mathbf{x}_{1:T}})$$

$$= \mathbb{I}_{\varepsilon>0} \cdot d_x \log(2\varepsilon) - \frac{1}{T}\mathrm{H}(p_{\mathbf{x}_{1:T}} \| \tilde{p}^\star_{\mathbf{x}_{1:T}})$$

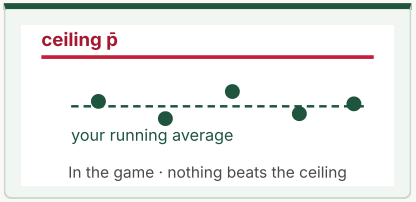


p̃* is the closest member of the convolution range R_ε; the attaining predictor is its deconvolution.

Hard part
Every pdf, every step — that is what the max runs over.

Tool
Box integral → convolution; the optimum projects onto the reachable set.

What it buys
A closed form — and the predictor that attains it.



C2 · The score of any continuous predictor

Take any continuous predictor, not the best one — its expected score is still pinned down up to $O(\epsilon)$.

Theorem 3 *(Approximation of the trajectory score · Auto. 2025, §V)*

For continuous densities, the score of a whole trajectory is pinned down by a **closed form**, up to $O(\epsilon)$:

$$\left| \bar{\mathcal{L}}_\epsilon(\hat{p}_{\mathbf{x}_{1:T}}, p_{\mathbf{x}_{1:T}}) - \mathbb{I}_{\epsilon>0}\, d_x \log(2\epsilon) + \frac{1}{T}\mathrm{H}(p_{\mathbf{x}_{1:T}} \| \hat{p}_{\mathbf{x}_{1:T}}) \right| = O(\epsilon)$$

- an $O(\epsilon)$ statement at **finite** ϵ — not an asymptotic one
- requires $p_{\mathbf{x}_{1:T}}$ and every $\hat{p}_{\mathbf{x}_k|\mathbf{x}_{1:k-1}}$ to be **continuous** ($\mathcal{P}_c$) — not "any predictor"

Hard part
No closed form for a given predictor's box integral.

Tool
ε-cells + continuity bound the error.

What it buys
O(ε) at finite ε — not asymptotic.



→ *But an expectation is something you never observe — what can one trajectory tell you?*

Yes — you only ever see one trajectory, and it still converges to the expected score.

Theorem 4 *(Convergence along a single trajectory · Auto. 2025, §V)*

Assume i.i.d. noise and a predictor that knows the dynamics: $\hat{p}_{\mathbf{x}_k|\mathbf{x}_{1:k-1}}(x_k \mid x_{1:k-1}) = \hat{p}_{\mathbf{w}}(x_k - f(x_{k-1}))$.

(i) The expectation collapses to one step:

$$\bar{\mathcal{L}}_\epsilon(\hat{p}_{\mathbf{x}_{1:T}}, p_{\mathbf{x}_{1:T}}) = \mathcal{L}_\epsilon(\hat{p}_{\mathbf{w}}, p_{\mathbf{w}})$$

(ii) One trajectory suffices — the path average converges in probability:

$$\lim_{T\to\infty} \bar{\mathcal{L}}_\epsilon(\hat{p}_{\mathbf{x}_{1:T}}, x_{1:T}) \overset{P}{=} \bar{\mathcal{L}}_\epsilon(\hat{p}_{\mathbf{x}_{1:T}}, p_{\mathbf{x}_{1:T}})$$

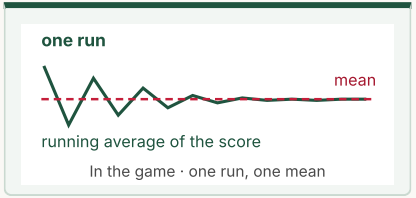
(iii) And at Monte-Carlo speed — provided $\mathbb{E}_{\mathbf{w}}\mathcal{L}_\epsilon(\hat{p}_{\mathbf{w}}, \mathbf{w})^2 < \infty$:

$$\Pr\left\{\left|\bar{\mathcal{L}}_\epsilon(\hat{p}_{\mathbf{x}_{1:T}}, x_{1:T}) - \bar{\mathcal{L}}_\epsilon(\hat{p}_{\mathbf{x}_{1:T}}, p_{\mathbf{x}_{1:T}})\right| \geq \delta\right\} = O\left(\frac{1}{\sqrt{T}}\right) \quad \forall\, \delta > 0$$

Hard part
Dependent samples — the expectation is invisible.

Tool
SLLN + Chebyshev on i.i.d. scores.

What it buys
$T^{-1/2}$ from one run, in probability.



→ *So what is a ceiling actually good for?*

Where the ceiling lives in ppsds

Every object on the ceiling has a name in `ppsds.predictability`.

on the ceiling	in `ppsds`
C1 the ceiling	`predictability.bounds.predictability_upper_bound(law_q, eps, d_x)`
C2 any predictor's score	`predictability.bounds.approximation_value(law_q, law_p, eps)`
C3 one trajectory	`predictability.trajectory.cumulative_average`, `across_trajectories`
the two entropies	`predictability.entropy.differential_entropy`, `cross_entropy`

- **Diagnose before you train** — will more data or more capacity help at all?
- **Benchmark honestly** — a defensible ceiling, not "we beat the baseline".
- **Design** — lower the noise entropy (better sensing) or coarsen the state; both raise the ceiling.

> *The library computes the bound, not the predictor that attains it: the ceiling is a reference line.*

In practice. The Huawei project I lead is putting this predictability tooling into production — happy to discuss after the talk.

What can you still guarantee?

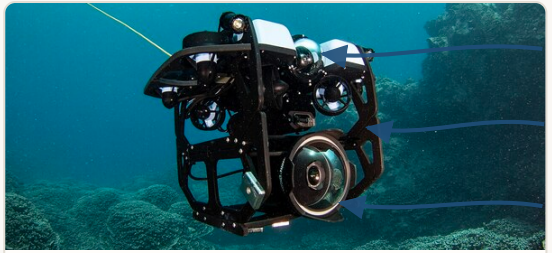
Part II — the floor

When the model is wrong

> **The predictor that is optimal for your model can fail badly on the real one.**



Aerial robot
kick: gust and slip



Underwater robot
kick: currents and turbulence



Planetary rover
kick: wheel slip on sand

- **The model's 90% region** misses where the gust puts it.
- **This is model error, not noise** — more data does not remove it.

IN THE GAME

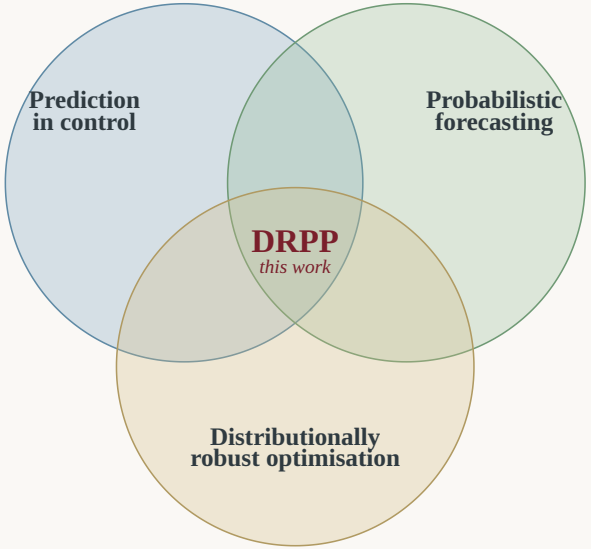
Neither the table's law nor the noise distribution is handed to you — you only see rounds, and you still have to place stakes. So you bet against a **set**, not against the true system.

Images: J. Sorenson, CC0 · Yoleeth, CC BY-SA 4.0 · NASA/JPL-Caltech, public domain.

→ *So how can a probabilistic predictor still guarantee performance when the SDS model is wrong?*

Where this sits

This work sits where three communities meet — and each leaves a different gap.



Control — pdfs from a nominal model.
Kalman 1960 · Sorenson 1971 · Julier & Uhlmann 1997
Gap: confident when the model shifts.

Forecasting — optimise a score on data.
Gneiting & Raftery 2007 · Koenker 2017 · Duan et al. 2020 · Salinas et al. 2020
Gap: no dynamics, so no guarantee under a model change.

DRO — optimise a decision under ambiguity.
Scarf 1957 · Gallego & Moon 1993 · Delage & Ye 2010 · Shapiro & Pichler 2024
Gap: the decision is Euclidean; here it is a pdf.

This work: a worst-case predictor over a set of dynamics, optimised in the space of predictive pdfs.

The plan

Formulate it, make it computable, then bracket the optimum.

FORMULATION · §IV

- **DRPP** — §IV-C · P_0
- **System & Predictor** — §IV-A · Eq. (2), Def. 1
- **Ambiguity Set** — §IV-B · Eq. (3)
- **Objective J** — §IV-C · Eq. (4)

→

METHODOLOGY · §V

- **Value Function V*** — §V-A · Eq. (5–6)
- **Bellman Equation** — §V-A · Thm 1
- **Dual & Optimality** — §V-B · D_1, Thm 2
- **One-step DRPP** — §V-C · P_1, P_2
- **Euclidean Minimax** — §V-C · Eq. (12)

→

SUBOPTIMAL PREDICTORS · §VI–VII

- **Noise-DRPP** — §VI-B · Thm 3
- **Upper Bound** — §VI-B · Thm 4
- **Eig-DRPP** — §VII-B · Thm 5
- **Lower Bound & Gap** — §VII-B · Thm 6

both branches relax one side — and both name their price

Experiments — does it hold in simulation?

→ *So what should we optimise for instead?*

> **Don't optimise for the model you have — optimise for the worst case you can't rule out.**

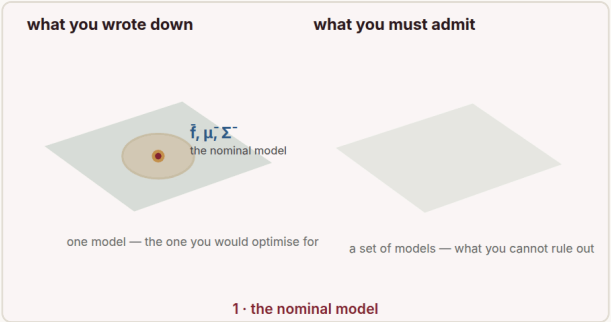
Definition 2 *(Conditional conic moment-based ambiguity set · TAC. 2026, §IV-B)*

Given a step k, a pair z, and nominal data $(\bar{f}_k, \bar{\mu}_k, \bar{\Sigma}_k)$, the set $\mathcal{I}_k(z)$ collects every model $\boldsymbol{x}_{k+1} = f_k(z) + \boldsymbol{w}_k$ you cannot rule out — three conic constraints, four knobs $\gamma_0, \ldots, \gamma_3$:

$$\text{dynamics: } \|f_k(z) - \bar{f}_k(z)\|_2^2 \le \gamma_0(z)$$

$$\text{noise mean: } \|\mathbb{E}(\boldsymbol{w}_k) - \bar{\mu}_k\|^2_{\bar{\Sigma}_k^{-1}} \le \gamma_1$$

$$\text{noise covariance: } \gamma_3\bar{\Sigma}_k \preceq \mathbb{E}(\boldsymbol{w}_k - \bar{\mu}_k)(\boldsymbol{w}_k - \bar{\mu}_k)^\top \preceq \gamma_2\bar{\Sigma}_k$$



γ_0 the table · γ_1 the mean · γ_2/γ_3 the spread · the set moves with z_k

You pay a little performance under the nominal model to buy a guarantee under all of them. Compared with the classical conic-moment set, γ_0 lets the dynamics be wrong too.

The predictor chooses a function; the adversary chooses a measure.

$$(\mathbf{P}_0): \quad \sup_{\mathscr{F}\in\mathfrak{F}} \inf_{\mathscr{P}\in\mathfrak{P}} J_0^{\mathscr{F},\mathscr{P}}(z_0), \qquad J_k^{\mathscr{F},\mathscr{P}}(z) := \mathbb{E}_{\mathscr{P}}\left[\sum_{t=k}^{T-1} \mathcal{L}\big(\mathscr{F}_t(\boldsymbol{z}_t), \boldsymbol{x}_{t+1}\big) \,\middle|\, \boldsymbol{z}_k = z\right]$$

You choose the predictor — a **function** from history to a predictive pdf.

The adversary chooses the system — a **probability measure** inside the ambiguity set.

outer sup over predictors · inner inf over systems — **both sides are infinite-dimensional.**

A detour that paid off. The same max–min idea led to initial results on a sixty-year-old problem — optimal mixed strategies in zero-sum differential games — and earned my 2025 NSFC doctoral grant.

→ *How do we solve a problem whose two sides are infinite-dimensional?*

The sequence problem collapses to a one-step recursion — exactly.

The first difficulty: $\mathbf{P}_0$ is an infinite-dimensional functional optimization — a predictor is a function, a system is a measure.

Theorem 1 *(Bellman equation · TAC. 2026, §V-A)*

With $V_T^*(z) = 0$, the robust value function obeys

$$V_k^*(z) = \sup_{\hat{p}_k \in \mathcal{F}} \inf_{\rho_k \in \mathcal{I}_k(z)} \int_{\mathcal{X}} \mathcal{L}(\hat{p}_k, x) + V_{k+1}^*\big(x, \pi_{k+1}(x)\big)\, \mathrm{d}\rho_k(x)$$

- an **exact** recursion, not a relaxation
- the inner problem is still over **probability measures**; making it finite-dimensional is a *separate* step

Recursion | Finite family | Euclidean | NP-hard

→ *But the inner problem is still over probability measures.*

The optimal predictor is forced into an exponential family — and at the terminal step, it is a Gaussian.

The route. Canonicalise the inner problem into an infinite-dimensional conic linear program → dualise it into $\mathbf{D}_1$ → its constraints are indexed by every $x \in \mathcal{X}$ → a **necessary** optimality condition replaces that infinity by a **finite family**.

Theorem 2 *(Optimality condition · TAC. 2026, §V-B)*

The optimal predictor lies almost everywhere in an exponential family:

$$\hat{p}_k(x) \overset{a.s.}{\propto} \exp\{-x^\top \theta_1 x - x^\top \theta_2 - V^*_{k+1}(x, \pi_{k+1}(x))\}, \qquad \theta_1 \in S^{d_x},\ \theta_2 \in \mathbb{R}^{d_x}$$

At the **terminal step** it collapses to a Gaussian:

$$\hat{p}_{T-1} \overset{a.s.}{\sim} \mathcal{N}(\hat{\mu}, \hat{\Sigma}) \qquad k = T-1$$

Recursion | **Finite family** | Euclidean | NP-hard

This is a necessary condition, not a characterisation of the multistep optimum.

→ *A parameterisation replaces a function — so the problem becomes finite.*

A Gaussian predictor is two parameters — and the one-step problem becomes finite-dimensional.

Start from the one-step problem $\mathbf{P}_1$, substitute the Gaussian $\hat{p}_k \sim \mathcal{N}(\hat{\mu}_k, \hat{\Sigma}_k)$, and two of the three variables are pinned down in closed form:

Lemma 1 *(Explicit solution of the one-step DRPP · TAC. 2026, §V-C)*

Given any mean μ_k, the worst-case covariance and the optimal predictive mean are

$$\Sigma_k^* = \gamma_2 \bar{\Sigma}_k - (\mu_k - \bar{\mu}_k)(\mu_k - \bar{\mu}_k)^\top, \qquad \hat{\mu}_k^* = \bar{\mu}_k + \bar{\nu}_k.$$

What remains is a **finite-dimensional** optimization over the matrix $\hat{\Sigma}_k$ and the two shifts $(a, b) = (\mu_k - \bar{\mu}_k,\ \nu_k - \bar{\nu}_k)$.

Recursion | Finite family | **Euclidean** | NP-hard

The first difficulty is resolved: a function became a few parameters.

→ *But the remaining problem is still hard.*

The problem is now finite-dimensional — and still NP-hard.

The one-step problem is now a **convex–nonconcave minimax**:

$$\min_{\hat{\Sigma}_k \succ 0} \max_{a,b} \; -\log\det(\hat{\Sigma}_k^{-1}) + \gamma_2 \bar{\Sigma}_k \cdot \hat{\Sigma}_k^{-1} + \|a+b\|^2_{\hat{\Sigma}_k^{-1}} - \|a\|^2_{\hat{\Sigma}_k^{-1}} \quad \text{s.t.} \quad \|a\|^2_{\bar{\Sigma}_k^{-1}} \le \gamma_1,\ \|b\|_2^2 \le \gamma_0(z)$$

- **Global hardness.** The inner problem is a convex maximisation — NP-hard in simple cases, e.g. quadratic maximisation over a hypercube [Pardalos 1988].
- **Local hardness.** A local minimax point may not exist [Jin et al. 2020]; gradient methods need exponentially many queries [Daskalakis et al. 2021]; a local stationary point is not always a local minimax point [Grimmer et al. 2022; Fiez et al. 2020].

Recursion · Finite family · Euclidean · **NP-hard**

This is the second difficulty: not how to compute the optimum, but how to relax it — and certify the gap.

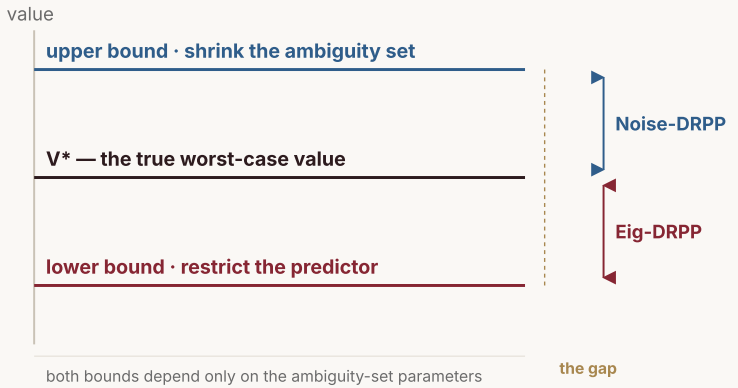
→ *So we stop chasing the optimum — and bracket it.*

Shrink the set for an upper bound; restrict the predictor for a lower bound.

Shrink the ambiguity set — remove the worst admissible systems. The inner $\inf$ can only rise, so the value can only rise: an **upper bound** on V^*.

Restrict the predictor — remove admissible bets. The outer $\sup$ can only fall, so the value can only fall: a **lower bound** on V^*.

Both bounds are computed from the ambiguity-set parameters alone — offline, before the run.



→ *The simplest bound: pretend the dynamics are exact.*

> **Trust the physics, doubt the noise — the optimal bet is a Gaussian with the nominal shape.**

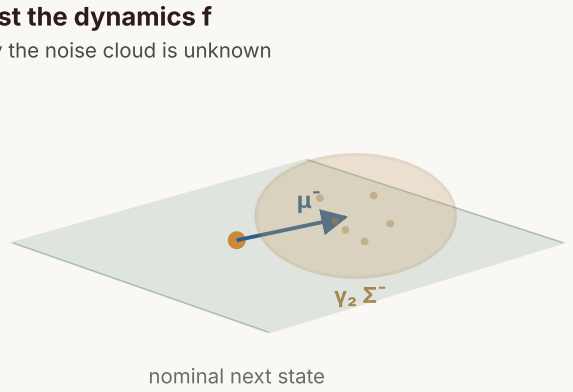
Theorem 3 *(Noise-DRPP · TAC. 2026, §VI-B)*

Assume the dynamics are **exact** ($\bar{\nu}_k = \nu_k$) and robust only to the noise. The optimal predictive pdf is

$$\hat{p}_k^* \sim \mathcal{N}\left(\bar{\nu}_k + \bar{\mu}_k,\ \gamma_2 \bar{\Sigma}_k\right).$$

Cost: none — nothing to optimise.

Center = nominal next state + nominal mean noise; covariance = $\gamma_2 \bar{\Sigma}_k$. The relaxed problem gives an **upper bound on** V^* (Thm 4 [TAC. 2026]), computable **offline**.



→ *Now doubt the physics too — and restrict how you bet.*

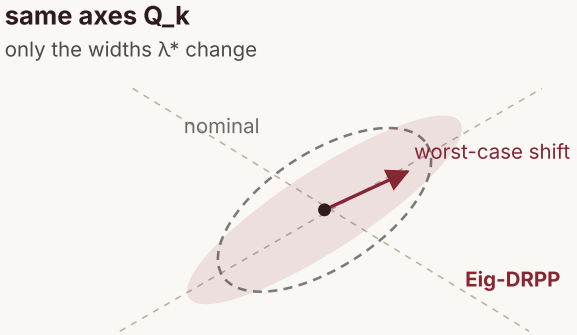
Doubt the physics, but keep the table's axes — the best bet is a Gaussian with adjusted widths.

Theorem 5 *(Eig-DRPP · TAC. 2026, §VII-B)*

Keep the **full ambiguity set** (including f) and instead constrain the predictive covariance to share the nominal eigenvectors ($\hat{Q}_k = Q_k$):

$$\hat{p}_k^* \sim \mathcal{N}\left(\bar{\nu}_k + \bar{\mu}_k,\ Q_k \hat{\Lambda}_k^* Q_k^\top\right).$$

Cost: d_x convex subproblems; take the best.



The axes stay the table's; only the widths $\hat{\Lambda}^*$ are chosen. This gives a **lower bound on** V^* (Thm 6 [TAC. 2026]) — provided $\gamma_0(z) \le \Gamma_0$ for all z.

→ *Both bounds are computable — so the distance between them is a price.*

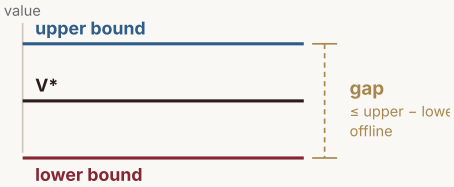
The gap is not solved away — it is bracketed, and priced.

Theorem 6 *(Optimality gap of Eig-DRPP · TAC. 2026, §VII-B)*

(ii) An upper bound on the gap, given $\gamma_0(z) \le \Gamma_0$:

$$V_k^*(z) - V_k^{\mathscr{F}_{\mathrm{Eig}}}(z) \le \sum_{t=k}^{T-1} -\frac{1}{2}\Bigg\{ \underbrace{d_x + \log\det(\gamma_2 \bar{\Sigma}_t)}_{\text{(a) from the upper bound}} - \underbrace{\sum_{i=1}^{d_x}\left[\log(\hat{\lambda}_{i,t}^*) + \gamma_2 \frac{\lambda_{i,t}}{\hat{\lambda}_{i,t}^*}\right] + \frac{2\sqrt{\Gamma_0\gamma_1\lambda_{j_t^*,t}} + \Gamma_0}{\hat{\lambda}_{j_t^*,t}^*}}_{\text{(b) from the lower bound}} \Bigg\}$$

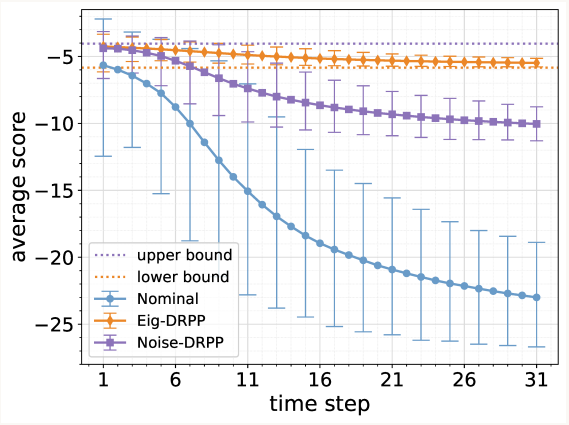
Their difference is an offline-computable **upper bound on the gap**: **(a)** from the upper bound, **(b)** from the lower bound. The gap becomes a **named price**, not an unknown defect.



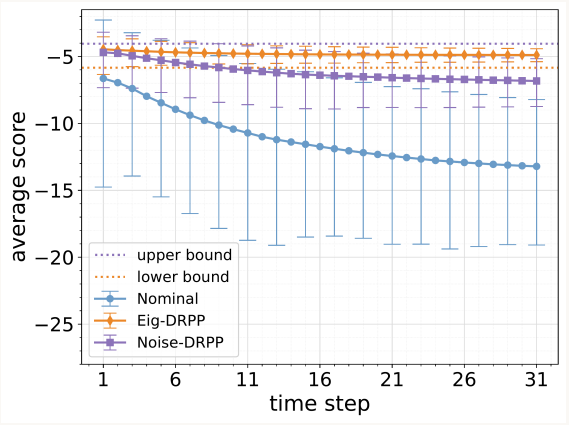
the second difficulty, bracketed

Experiments · Does it hold in simulation?

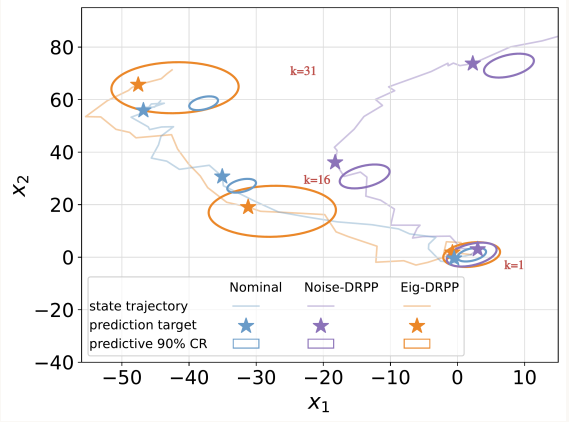
Under adversarial dynamics the nominal predictor collapses — the Noise-DRPP / Eig-DRPP predictors hold.



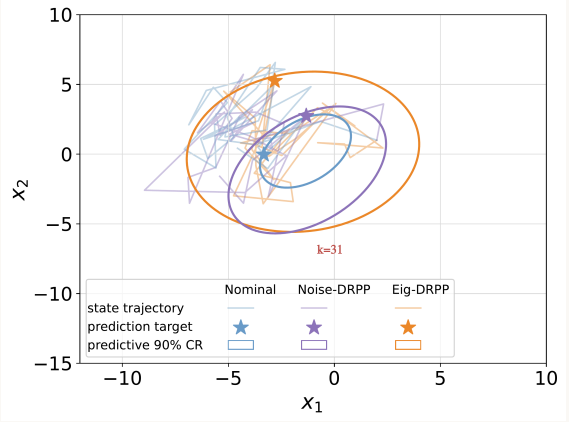
Adversarial, zero control — nominal collapses.



Adversarial, LQR — same story.



90% regions, automotive — the star misses the nominal ellipse.



90% regions, LQR — same.

Where the floor lives in ppsds

> **Every step of the floor has an object in `ppsds` — and each family of ambiguity sets yields its own DRPP.**

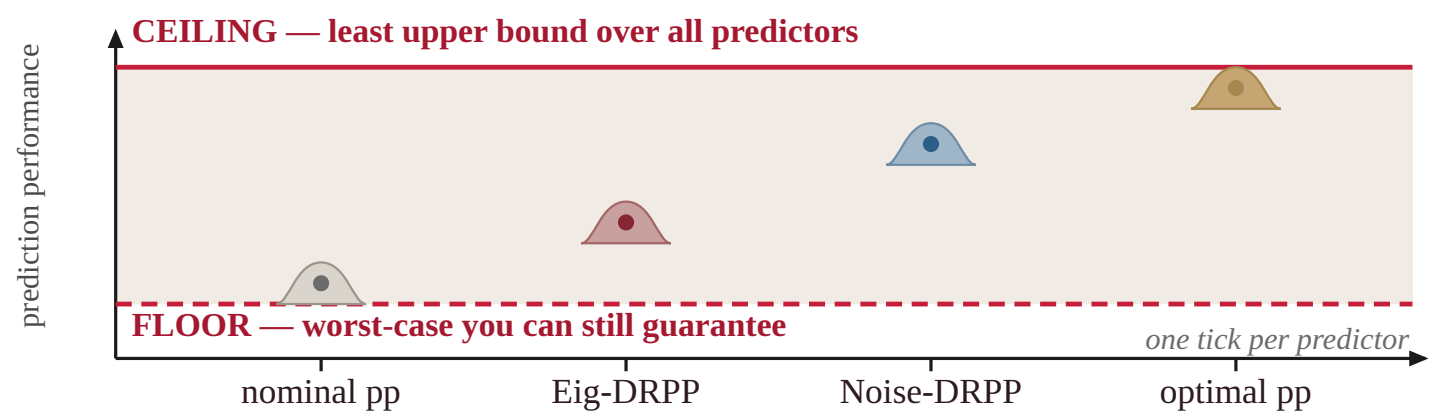
on the floor	in `ppsds`
Formulation · the ambiguity set	`ambiguity.additive_drift.conic_moment.LinearAmbiguitySet`, `NonlinearAmbiguitySet`
Formulation · the objective	the maximin objective — no object of its own
Methodology · the worst case, per step	`probabilistic_predictor.drpp...eigvec_fixed_minmax`
Suboptimal predictors · the two predictors	`NominalPredictor`, `NoiseDrpp`, `EigDrpp`
Suboptimal predictors · the gap	a paper result (Thm 6); the library reports the pieces it is computed from

> *A new family is a new module beside the existing one, never a branch inside it:* `ambiguity/`, `probabilistic_predictor/drpp/`, `environment/dynamics_mode/adversarial/`, `tests/` *— four places, one family.*

Putting it together

The ceiling, the floor, and what is between

The band



Every predictor lives between a ceiling it cannot beat and a floor it should not cross.

- The **ceiling** [Auto. 2025] bounds every predictor from above; the **floor** [TAC. 2026] is what you can still guarantee when the model is wrong.

→ *So what lives in the band between them?*

Closing the gap

> **The remaining question is concrete: use more information to narrow the gap between the two bounds.**

- **More information** — context, inputs, trajectory data: raise the ceiling by seeing more, or approach the floor by knowing more accurately.
- **Weaker assumptions** — nonlinear and learning-based predictors; sets estimated from finite samples.
- **Higher dimension** — the results are explicit in d_x, but high-dimensional systems are untested.

> *None of these three is closed — if you are working on one of them, come and find me after the talk.*

If asked — where else? Training-cost analysis, differential privacy (p9), and safety assessment.

You cannot beat the ceiling. You should not cross the floor. What remains is how to close the gap with more information.

What you take home

> **A ceiling you can compute, a floor you can certify — and an implementation that ships both.**

1. **A ceiling** — compute it before promising gains.
2. **A floor** — declare the set, then certify.
3. **An implementation** — `ppsds` ships both; every number is computed, not copied.

Automatica, 2025 — Xu, Li, He · 10.1016/j.automatica.2025.112160

IEEE TAC, 2026 — Xu, He · 10.1109/TAC.2026.3687495

github.com/Xuzerken/ppsds · tao-xu.com/ppsds/

```
pip install ppsds==0.0.3a0
```

> *I am devoted to understanding, overcoming, exploiting uncertainties in dynamical systems.*

Homepage
tao-xu.com

Thank you. Welcome to collaborate — let's talk.

Play on your phone

> **The game is live — play a round while we take questions.**



Play on your phone

www.tao-xu.com/ppsds/examples/understand_ppsds_by_pmf_betting/game/

```
pip install ppsds==0.0.3a0
```

Questions? Zerken@sjtu.edu.cn

Appendix — the library

Q&A backup — install, entry points, and what the library is for

What it is, and one install

> **One install gives you both sides: the reference line and the predictor.**

```
pip install ppsds==0.0.3a0      # the alpha is a pre-release: pin it, or pass --pre
```

- **Core** needs only numpy, scipy, cvxpy and gymnasium — Python 3.10 or newer.
- `ppsds.predictability` computes the ceiling; it needs no predictor at all.
- `ppsds.probabilistic_predictor` gives you the floor: nominal, Noise-DRPP and Eig-DRPP.
- **Extras** (`ppsds[experiments]`) add the batch runners and the paper figures.

Who it is for

Four people come to this library, and they need four different things.

C1 **You have a system you can step.**

Put a predictor on it, and know when it will lie to you. This is the main path.

C2 **You want to do research in this direction.**

Add a family of ambiguity sets, or change the assumptions — the interface is the contribution.

C3 **You want to check or reproduce a paper.**

Equation numbers trace to code, and the figures are recomputed, not copied.

E1 **You are just curious.**

Thirty seconds on the table, no install — the game you saw at the start.

Where you start, how far you go

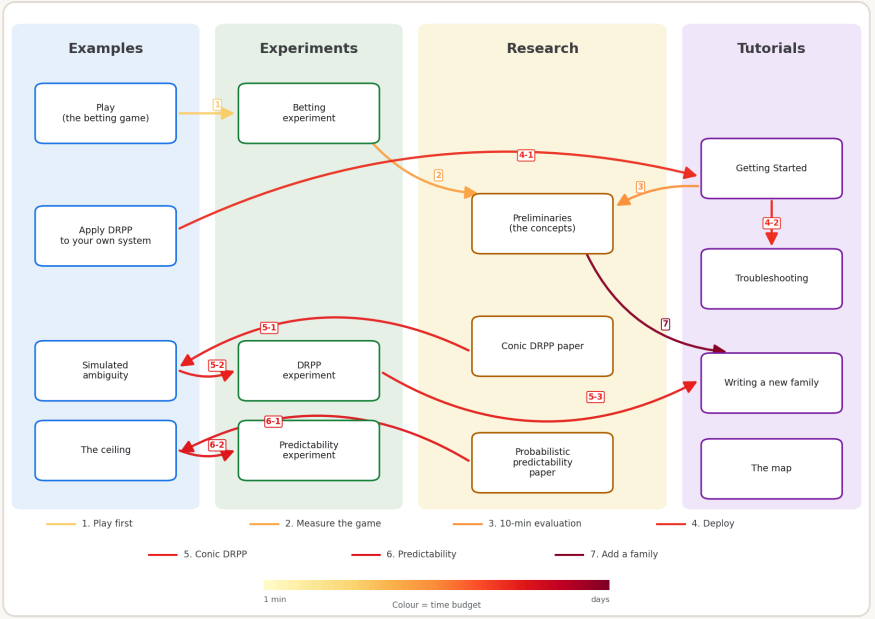
> **Rows are how far you want to go; columns are what you start from.**

	A · Just curious	B · Build it yourself	C · Your system	D · Data only
L0 · See	the game	—	—	—
L1 · Use	—	reproduce a paper	put DRPP on your environment	read the limits first
L2 · Extend	—	add a predictor kind	add a family	—
L3 · Research	—	work on the ceiling	research on the floor	—
L4 · Contribute	report a problem	contribute a result	contribute a result	discuss a collaboration

The README uses the same table.

Several ways in, ordered by what you already have.

1. **Play first.** The game → the betting experiment → preliminaries. Nothing to install.
2. **Deploy on your system.** The apply example → Getting Started → Troubleshooting: the four-step declaration, the pitfalls, the two ways it surprises you.
3. **Check or reproduce a paper.** The paper page → the example → the experiment: six golden numbers, then the figure at paper scale.
4. **Add a family.** The preliminaries → either paper → the developer guide → CONTRIBUTING. This route ends in becoming a contributor.



The same routes on the site's `map` page.

Recall: the theory as code

> **Everything from the last three parts has a name in the package.**

you saw	in `ppsds`
C1 the ceiling	`predictability.bounds.predictability_upper_bound`
C2 any predictor's score	`predictability.bounds.approximation_value`
C3 one trajectory	`predictability.trajectory.cumulative_average`
Formulation the ambiguity set	`ambiguity.additive_drift.conic_moment.LinearAmbiguitySet` · `NonlinearAmbiguitySet`
Methodology + suboptimal predictors the worst case and the predictors	`eigvec_fixed_minmax` · `NominalPredictor` · `NoiseDrpp` · `EigDrpp`
the game's bets	`numerics.atom_pmf.AtomPmf` · `DriftShiftedAtomPmfPredictor`

> *The point of the package is not new mathematics — it is that each piece above is an object you can import and run.*

Declare first, predict second; swapping them is the usual mistake.

1 · the model	2 · the family	3 · its parameters	4 · the predictors
nominal drift and noise	conic-moment, additive drift	the four radii	nominal, Noise-DRPP, Eig-DRPP

On a two-state stand-in, 50 steps: **nominal −2.867, Eig-DRPP −3.366** — and nominal winning is the expected outcome, not a failure.

> *The loop is short; the declaration is the work.*

Two surprises, and when it pays

The two things that surprise users are both about the declaration, not the model.

Model error is not why DRPP wins

Acrobot: the linear residual is **74%** of the state increment, and with 20 identification episodes DRPP still loses.

identification data	nominal	Eig-DRPP	winner
1 episode (covariance 3.9× too small)	−6.062	−2.515	DRPP
20 episodes	−1.143	−7.099	nominal

The covariance may be unusable

Pendulum: condition number 7.04×10^{11} and one numerically zero eigenvalue. Unregularised, the score is $-\infty$.

regularization floor	nominal	Eig-DRPP
10^{-6}	9.164	3.659
10^{-2}	3.711	2.851

Moving the floor from 10^{-6} to 10^{-2} moves the score by **5.5 points**: the floor is a modelling claim.

> *Too little identification data is the most common source of optimism — and optimism, not model error, is what DRPP is paid for.*

The cost and the boundaries

The floor covers exactly the systems you declared — no more.

- **Conditional.** If the true system is outside the set, the guarantee is vacuous.
- **Too wide costs average score**; too narrow voids the guarantee and looks like failure.
- **The set must contain its own generator** — a simulated truth outside it is a declaration bug.
- **Conic-moment covers additive drift error and additive, homogeneous noise** — not state-dependent or multiplicative noise.
- **No predictor attains the ceiling:** it is a reference line, not a target.

> `diagnose()` *describes your residuals; it cannot tell whether your set contains the real system.*

Welcome: contribute, collaborate, discuss

> **A research package's most valuable contribution is the one that keeps a number stable.**

1. **Played the game and found something wrong?** Open an issue — the game is a real user of the library, so that report counts.

2. **Built a simulation or used your own system?** File an L1 issue, or send an L2/L3 result (including a benchmark) as a pull request.

3. **Work on a system this family cannot express?** Discuss a new family — the useful next step is usually a contract, not a full implementation.

4. **Want to do research together?** The "data only" column is deliberately future research; a discussion is the right first contribution.

> *A change that touches numerics must survive an A/B comparison — bit-identical, not "close".*

Six steps, and you have seen what a probabilistic prediction is.

1 Prediction on



→

2 Place a stack



→

3 Cover the support



↓

4 Set a height



←

5 Deal



←

6 Next round



Play on your phone
www.tao-xu.com/ppsds/examples/understand_ppsds_by_pmf_betting/game/