Optimal and Learning Control for Autonomous Robots

Lecture 10

ADRL

Jonas Buchli/ Farbod Farshidian
Agile & Dexterous Robotics Lab

FNSNF

robotics Swiss National Centre of Competence in Research

ETH Zürich

Reading

- Peters, Jan, and Stefan Schaal. "Reinforcement learning of motor skills with policy gradients."
- Deisenroth, Marc Peter, Gerhard Neumann, and Jan Peters. "A Survey on Policy Search for Robotics." (2013). [Section 2.2]

Outline

- Natural Gradient
- episodic Natural Actor Critic (eNAC)

Policy Gradient Theorem (PGT)

- Gradient in Policy Gradient Theorem (PGT)

$$\nabla_\theta^{PGT} J(\theta) = E_{p_\theta(\tau)}\left[\sum_{t=0}^{T-1} \nabla_\theta \left(\log \pi_\theta(u_t \mid x_t)\right)\left(Q_t^\pi(x_t,u_t) - b_t\right)\right]$$

- If $b_t = V_t^\pi(x_t)$

$$\nabla_\theta^{PGT} J(\theta) = E_{p_\theta(\tau)}\left[\sum_{t=0}^{T-1} \nabla_\theta \left(\log \pi_\theta(u_t \mid x_t)\right)\left(Q_t^\pi(x_t,u_t) - V_t^\pi(x_t)\right)\right]$$

Advantage function $A_t^\pi(x_t,u_t) = Q_t^\pi(x_t,u_t) - V_t^\pi(x_t)$

$$\nabla_\theta^{PGT} J(\theta) = E_{p_\theta(\tau)}\left[\sum_{t=0}^{T-1} \nabla_\theta \left(\log \pi_\theta(u_t \mid x_t)\right) A_t^\pi(x_t,u_t)\right]$$

Unbiased estimation of gradient

- We need to approximate the advantage function

$$\nabla_\theta^{PGT} J(\theta) = E_{p_\theta(\tau)}\left[\sum_{t=0}^{T-1} \nabla_\theta \left(\log \pi_\theta(u_t \mid x_t)\right) A_t^\pi(x_t, u_t)\right]$$

- The approximation is done through function approximation
- This function approximation should not cause bias in the gradient estimation
- But every function approximation has error
- So the error should be orthogonal to the gradient direction

Compatible function approximation

- The function approximation for the advantage function should minimize the expectation of the square error (ESE)

$$\min_{w} E_{p_\theta}\left[\left(A_t^{\pi}(x_t,u_t) - f_w(x_t,u_t)\right)^2\right]$$

- The function approximation is linear with respect to its parameters $f_w(x_t,u_t) = w^T \underbrace{\nabla_\theta\left(\log \pi_\theta(u_t \mid x_t)\right)}_{\text{Base functions are gradient of policy}}$

- *w* should be found through minimizing the ESE

PGT with compatible function approximation

- Using the function approximation in the PGT gradient <u>while the baseline is taken as value function</u>

$$\nabla_\theta^{PGT} J(\theta) = G_\theta w$$

$$G_\theta = E_{p_\theta(\tau)}\left[\sum_{t=0}^{T} \nabla_\theta\left(\log \pi_\theta(\mathrm{u}_t \mid \mathrm{x}_t)\right)\nabla_\theta\left(\log \pi_\theta(\mathrm{u}_t \mid \mathrm{x}_t)\right)^T\right]$$

Goals of Natural Gradient

- Avoiding quick decrease in the exploration ability
- keeping the exploitation of the gradient information local

Idea of Natural Gradient

limit the changes of the policy distribution or equivalently the changes of trajectory distribution

Natural Gradient

So the problem statement is:

Find the parameter change which maximize the cost function below while keeping distance between two distributions ε

$$\max_{\Delta\theta} J(\theta+\Delta\theta) \approx J(\theta) + \Delta\theta^T \nabla_\theta J$$

$$s.t. \quad \varepsilon = d_{KL}\left(p_\theta(\tau)\, \mathrm{P}\, p_{\theta+\Delta\theta}(\tau)\right) \approx \frac{1}{2}\Delta\theta^T F_\theta \Delta\theta$$

Learning rate

$$\Delta\theta = \alpha_n F_\theta^{-1} \nabla_\theta J$$

$$F_\theta = E_{p_\theta(\tau)}\left[\sum_{t=0}^{T} \nabla_\theta \left(\log \pi_\theta(\mathrm{u}_t \mid \mathrm{x}_t)\right) \nabla_\theta \left(\log \pi_\theta(\mathrm{u}_t \mid \mathrm{x}_t)\right)^T\right]$$

Natural Policy Gradient

Using PGT gradient in natural gradient format

$$\nabla_\theta^{NG} J(\theta) = F_\theta^{-1} \nabla_\theta^{PG} J(\theta)$$

$$\nabla_\theta^{PGT} J(\theta) = G_\theta w$$

From PGT

$$\nabla_\theta^{NG} J(\theta) = F_\theta^{-1} G_\theta w$$

$$G_\theta = E_{p_\theta(\tau)}\left[\sum_{t=0}^{T} \nabla_\theta \left(\log \pi_\theta(\mathrm{u}_t \mid \mathrm{x}_t)\right) \nabla_\theta \left(\log \pi_\theta(\mathrm{u}_t \mid \mathrm{x}_t)\right)^T\right]$$

$$F_\theta = E_{p_\theta(\tau)}\left[\sum_{t=0}^{T} \nabla_\theta \left(\log \pi_\theta(\mathrm{u}_t \mid \mathrm{x}_t)\right) \nabla_\theta \left(\log \pi_\theta(\mathrm{u}_t \mid \mathrm{x}_t)\right)^T\right]$$

$$F_\theta = G_\theta$$

$$\nabla_\theta^{NG} J(\theta) = w$$

Natural Policy Gradient plus PGT with value function baseline

- Using PGT gradient with value function baseline in natural gradient format yields

$$\nabla_\theta^{NG} J(\theta) = w$$

- We just need to compute *w* through minimizing ESE

$$\min_w E_{p_\theta}\left[\left(A_t^\pi(x_t,u_t) - w^T \nabla_\theta \log \pi_\theta(u_t \mid x_t)\right)^2\right]$$

ADRL

ETH Zürich

Approximating advantage function

- To solve ESE, we should have the advantage function of the current policy.

$$\min_{w} E_{p_\theta}\left[\left(A_t^\pi(x_t,u_t) - w^T\nabla_\theta \log \pi_\theta(u_t \mid x_t)\right)^2\right]$$

- But we don't have the advantage function explicitly

Can we compute an estimation of advantage function?

Advantage function

- From definition of advantage function we have

$$Q_t^{\pi}(x_t, u_t) = A_t^{\pi}(x_t, u_t) + V_t^{\pi}(x_t)$$

- From definition of state-action value function we have

$$Q_t^{\pi}(x_t, u_t) = r_t(x_t, u_t) + \int V_{t+1}^{\pi}(x')p(x' \mid x_t, u_t)dx'$$

- Combining these two formulas

$$A_t^{\pi}(x_t, u_t) + V_t^{\pi}(x_t) = r_t(x_t, u_t) + \int V_{t+1}^{\pi}(x')p(x' \mid x_t, u_t)dx'$$

Advantage function estimation

- We got a Bellman like equation for the advantage function:

$$A_t^\pi(x_t,u_t)+V_t^\pi(x_t)=r_t(x_t,u_t)+\int V_{t+1}^\pi(x')p(x'\mid x_t,u_t)dx'$$

- For the samples derived form rollout, we can write

$$\underbrace{\tilde{A}_t^\pi(x_t,u_t)+\tilde{V}_t^\pi(x_t)}_{\text{Estimated values}}=r_t(x_t,u_t)+\underbrace{\tilde{V}_{t+1}^\pi(x_{t+1})}_{\text{Estimated values}}+\underbrace{\varepsilon_t}_{\text{error}}$$

For every time step *t*

Rollout: the trajectory of state and actions yields from execution of policy in the environment

$$\tau: x_0,u_0,x_1,u_1,\ldots,x_t,u_t,x_{t+1},u_{t+1},\ldots,x_{H-1},u_{H-1},x_H$$

episodic Natural Actor Critic (eNAC)

- Use the estimated advantage function in each time step and sum them up

$$\tilde{A}_0^{\pi}(x_0,u_0)+\tilde{V}_0^{\pi}(x_0)=r_0(x_0,u_0)+\cancel{\tilde{V}_1^{\pi}(x_1)}+\varepsilon_0$$

$$\tilde{A}_1^{\pi}(x_1,u_1)+\cancel{\tilde{V}_1^{\pi}(x_1)}=r_1(x_1,u_1)+\cancel{\tilde{V}_2^{\pi}(x_2)}+\varepsilon_1$$

$$\vdots$$

$$+$$

$$\tilde{A}_t^{\pi}(x_t,u_t)+\cancel{\tilde{V}_t^{\pi}(x_t)}=r_t(x_t,u_t)+\cancel{\tilde{V}_{t+1}^{\pi}(x_{t+1})}+\varepsilon_t$$

$$\tilde{A}_{t+1}^{\pi}(x_{t+1},u_{t+1})+\cancel{\tilde{V}_{t+1}^{\pi}(x_{t+1})}=r_{t+1}(x_{t+1},u_{t+1})+\cancel{\tilde{V}_{t+2}^{\pi}(x_{t+2})}+\varepsilon_{t+1}$$

$$\vdots$$

$$\tilde{A}_{T-1}^{\pi}(x_T,u_T)+\cancel{\tilde{V}_T^{\pi}(x_T)}=r_T(x_T)+\varepsilon_T$$

$$\sum_{t=0}^{T-1}\tilde{A}_t^{\pi}(x_t,u_t)+\tilde{V}_0^{\pi}(x_0)=\sum_{t=0}^{T}r_t(x_t,u_t)+\sum_{t=0}^{T}\varepsilon_t$$

Continued

- Using function approximation for advantage function

$$\sum_{t=0}^{T-1} \tilde{A}_t^{\pi}(x_t,u_t) + \tilde{V}_0^{\pi}(x_t) = \sum_{t=0}^{T} r_t(x_t,u_t) + \sum_{t=0}^{T} \varepsilon_t \qquad \tilde{A}_t^{\pi}(x_t,u_t) \approx w^T \nabla_\theta \log \pi_\theta(u_t \mid x_t)$$

$$\Downarrow$$

$$\sum_{t=0}^{T-1} w^T \nabla_\theta \log \pi_\theta(u_t \mid x_t) + \tilde{V}_0^{\pi}(x_0) = \sum_{t=0}^{T} r_t(x_t,u_t) + \sum_{t=0}^{T} \varepsilon_t$$

$$w^T \sum_{t=0}^{T-1} \nabla_\theta \log \pi_\theta(u_t \mid x_t) + \tilde{V}_0^{\pi}(x_0) = \sum_{t=0}^{T} r_t(x_t,u_t) + \sum_{t=0}^{T} \varepsilon_t$$

Continued

- Now what about the value function for initial time

$$w^T \sum_{t=0}^{T-1} \nabla_\theta \log \pi_\theta(u_t \mid x_t) + \boxed{\tilde{V}_0^\pi(x_0)} = \sum_{t=0}^{T} r_t(x_t, u_t) + \sum_{t=0}^{T} \varepsilon_t$$

- We need to approximate the initial value function as well

$$\tilde{V}_0^\pi(x_0) \approx v^T \varphi(x_0)$$

  - If the agent is always initialized in a specific state, the base function is simply one and v is accumulated reward of the trajectory
  - If the agent is initialized in random states, the base function should be a function of state vector

Continued

- Using function approximation for value function

$$w^T \sum_{t=0}^{T-1} \nabla_\theta \log \pi_\theta(u_t \mid x_t) + \nu^T \varphi(x_0) = \sum_{t=0}^{T} r_t(x_t, u_t) + \sum_{t=0}^{T} \varepsilon_t$$

- In the vector form we can write

$$\begin{bmatrix} w \\ \nu \end{bmatrix}^T \begin{bmatrix} \sum_{t=0}^{T-1} \nabla_\theta \log \pi_\theta(u_t \mid x_t) \\ \varphi(x_0) \end{bmatrix} = \sum_{t=0}^{T} r_t(x_t, u_t) + \sum_{t=0}^{T} \varepsilon_t$$

- To abbreviate the notation

$$\underbrace{\begin{bmatrix} w \\ \nu \end{bmatrix}^T}_{\text{Parameter vector}} \underbrace{\begin{bmatrix} \phi \\ \varphi(x_0) \end{bmatrix}}_{\text{Base function}} = R + \varepsilon$$

Error of Estimation & Approximation

acc. reward of trajectory

$$\phi = \sum_{t=0}^{T-1} \nabla_\theta \log \pi_\theta(u_t \mid x_t)$$

$$R = \sum_{t=0}^{T} r_t(x_t, u_t)$$

$$\varepsilon = \sum_{t=0}^{T} \varepsilon_t$$

eNAC Algorithm

- To reduce the error, we should use information from several rollouts (say N rollouts)

$$\begin{bmatrix} \phi^{1^T} & \varphi(x_0^1)^T \end{bmatrix} \begin{bmatrix} w \\ v \end{bmatrix} = R^1 + \varepsilon^1$$

$$\vdots$$

$$\begin{bmatrix} \phi^{i^T} & \varphi(x_0^i)^T \end{bmatrix} \begin{bmatrix} w \\ v \end{bmatrix} = R^i + \varepsilon^i$$

$$\vdots$$

$$\begin{bmatrix} \phi^{N^T} & \varphi(x_0^N)^T \end{bmatrix} \begin{bmatrix} w \\ v \end{bmatrix} = R^N + \varepsilon^N$$

Use Least Square methods to estimate the parameter vector

eNAC algorithm

- Using the least square method over *N* rollouts, we will have

$$\Psi = \begin{bmatrix} \Phi^{1^T} \\ \vdots \\ \Phi^{N^T} \end{bmatrix}, \quad \Phi^i = \begin{bmatrix} \sum_{t=0}^{T-1} \nabla_\theta \log \pi_\theta(u_t^i \mid x_t^i) \\ \varphi(x_0^i) \end{bmatrix} \qquad R = \begin{bmatrix} R^1 \\ \vdots \\ R^N \end{bmatrix}, \quad R^i = \sum_{t=0}^{T} r_t(x_t^i, u_t^i)$$

$$\boxed{\begin{bmatrix} w \\ v \end{bmatrix} = \left(\Psi^T \Psi\right)^{-1} \Psi^T R}$$

Algorithm Episodic Natural Actor Critic

Input: Policy parametrization $\boldsymbol{\theta}$,

data-set $\mathcal{D} = \left\{ \boldsymbol{x}_{1:T}^{[i]}, \boldsymbol{u}_{1:T-1}^{[i]}, r_{1:T}^{[i]} \right\}_{i=1\ldots N}$

for each sample $i = 1 \ldots N$ **do**

Compute returns: $R^{[i]} = \sum_{t=0}^{T} r_t^{[i]}$

Compute features: $\boldsymbol{\psi}^{[i]} = \begin{bmatrix} \sum_{t=0}^{T-1} \nabla_{\boldsymbol{\theta}} \log \pi_{\boldsymbol{\theta}} \left(\boldsymbol{u}_t^{[i]} \middle| \boldsymbol{x}_t^{[i]}, t \right) \\ \varphi(\boldsymbol{x}_0^{[i]}) \end{bmatrix}$

end for

Fit advantage function and initial value function

$$\boldsymbol{R} = \left[R^{[1]}, \ldots, R^{[N]} \right]^T, \quad \boldsymbol{\Psi} = \left[\boldsymbol{\psi}^{[1]}, \ldots, \boldsymbol{\psi}^{[N]} \right]^T$$

$$\begin{bmatrix} \boldsymbol{w} \\ \boldsymbol{v} \end{bmatrix} = (\boldsymbol{\Psi}^T \boldsymbol{\Psi})^{-1} \boldsymbol{\Psi}^T \boldsymbol{R}$$

return $\nabla_{\boldsymbol{\theta}}^{\mathrm{eNAC}} J_{\boldsymbol{\theta}} = \boldsymbol{w}$

Credits

material from:

Reinforcement Learning of motor skill with policy gradients

A Survey on Policy Search for Robotics

19.11.2013

Optimal and Learning Control

for

Autonomous Robots

Lecture 10

Jonas Buchli
Agile & Dexterous Robotics Lab

Exercise 3

online until end of week!

Office hour

No office hour this week!

Next week, two office hours:
Thu 13h-15, 17:30-18:30

Next week's lecture
RL Recap

Reading

There is no book!

Learning variable impedance control.
Buchli, Stulp, Theodorou, Schaal, IJRR 30(7), 820-33

Outline

Natural Gradient
Natural Actor Critic

Path Integral Stochastic Optimal Control
Policy Improvements with Path Integrals

eNAC

Avoid calculating explicit gradient?

Idea: Modify current best guess for optimal controls - Pick best seen outcome as new best guess

- This works both as global, one step algorithm
- and local, iterative algorithm
- one step algorithms run into curse of dimensionality, iterative work in practice but give local optimum
- No need for step-size! Complete update step extracted from data!

PI^2

PoWER

ADRL

Stochastic optimal control

$$R(\tau_i) = \phi_{t_N} + \int_{t_i}^{t_N} r_t \, dt$$

$$r_t = r(\mathbf{x}_t, \mathbf{u}_t, t) = q_t + \frac{1}{2}\mathbf{u}_t^T \mathbf{R} \mathbf{u}_t$$

q arbitrary function of x,t (but not u)

state dependent input gain matrix

nonlinear system dynamics

Linear in controls and noise

$$\dot{\mathbf{x}}_t = \mathbf{f}(\mathbf{x}_t, t) + \mathbf{G}(\mathbf{x}_t)(\mathbf{u}_t + \varepsilon_t)$$

Noise: meanfree, gaussian

$$\mathbf{f}_t + \mathbf{G}_t(\mathbf{u}_t + \varepsilon_t)$$

$$V(x_t) = \min_{u_t} E_\tau[R(\tau)]$$

$$u^*(x, t) = \underset{u_t}{\operatorname{argmin}} E_\tau[R(\tau)]$$

Lecture 3: LQR

quadratic cost

$$V = \tfrac{1}{2}\Delta\mathbf{x}^T(t_f)\phi_{\mathbf{xx}}(t_f)\Delta\mathbf{x}(t_f) + \frac{1}{2}\int_{t_0}^{t_f}\left\{[\Delta\mathbf{x}^T(t)\ \Delta\mathbf{u}^T(t)]\begin{bmatrix}\mathbf{Q}(t) & \mathbf{M}(t)\\ \mathbf{M}^T(t) & \mathbf{R}(t)\end{bmatrix}\begin{bmatrix}\Delta\mathbf{x}(t)\\ \Delta\mathbf{u}(t)\end{bmatrix}\right.$$

linear dynamics

$$\Delta\dot{\mathbf{x}}(t) = \mathbf{F}(t)\Delta\mathbf{x}(t) + \mathbf{G}(t)\Delta\mathbf{u}(t),$$

ADRL

HJB equation

Solving ~~Quadratic~~ HJB Equation

LHS

$$\frac{\partial V^*}{\partial t}[\Delta \mathbf{x}^*(t), t] = -\min_{\mathbf{u}} \mathscr{H}[\Delta \mathbf{x}^*(t), \Delta \mathbf{u}(t), t]$$

0

$$\cancel{\tfrac{1}{2}\Delta \mathbf{x}^{*T}(t)\dot{\mathbf{P}}(t)\Delta \mathbf{x}^*(t)}$$

RHS

1

$$\frac{\partial}{\partial \mathbf{u}}(.) = 0$$

3

$$\mathbf{u}^*$$

4

~~2~~

$$\cancel{\frac{\partial V^*}{\partial \Delta \mathbf{x}}[\Delta \mathbf{x}^*(t), t] = \Delta \mathbf{x}^{*T}(t)\mathbf{P}(t)}$$

Find Value Function and Optimal Controls

Find right side of HJB:

1. by partial derivative with respect to controls and setting it to 0
2. ~~Use partial derivative of Quadratic Ansatz to substitute partial derivative of V in respect to x~~
3. solve in this expression for u: yields u* (optimal control!)
4. substitute u* back into HJB, ~~solve for unknown matrix P~~

A D R L

ETH Zürich

$$V(x_t) = \min_{u_t} E_\tau [R(\tau)]$$

$$\int p(\tau) R(\tau) d\tau \qquad \int p(\tau) \left(\frac{1}{\lambda}\phi + \frac{1}{\lambda} \int r dt \right) d\tau$$

Discretize and use EM-like idea: PoWER -
Problem: pseudo-probability - restriction on cost function

Other idea: Treat probability as a diffusion process - Connection with statistical physics
Forward dynamics! Sampling (Monte Carlo)!

Derivation of stochastic HJB

Stochastic principle of optimality

Principle of optimality

$$V^*(t_1) = E\left\{ \phi[\mathbf{x}^*(t_f), t_f] - \int_{t_f}^{t_1} \mathscr{L}[\mathbf{x}^*(t), \mathbf{u}^*(t), t]\, dt \right\}$$

Total time derivative:

$$\frac{dV^*(t_1)}{dt} = -E\{\mathscr{L}[\mathbf{x}^*(t_1), \mathbf{u}^*(t_1), t_1]\}$$

Measurements are deterministic

$$\frac{dV^*(t_1)}{dt} = -\mathscr{L}[\mathbf{x}^*(t_1), \mathbf{u}^*(t_1), t_1]$$

[St] p 422/23

Can also write total time derivative as Taylor Series

$$\frac{dV^*}{dt}\Delta t = E\left\{\frac{\partial V^*}{\partial t}\Delta t + \frac{\partial V^*}{\partial \mathbf{x}}\dot{\mathbf{x}}\,\Delta t + \tfrac{1}{2}\left[\dot{\mathbf{x}}^T \frac{\partial^2 V^*}{\partial \mathbf{x}^2}\dot{\mathbf{x}}\right]\Delta t^2 + \cdots\right\}$$

$$= E[\,V_t^*\,\Delta t + V_{\mathbf{x}}^*(\mathbf{f} + \mathbf{L}\mathbf{w})\,\Delta t + \tfrac{1}{2}(\mathbf{f} + \mathbf{L}\mathbf{w})^T V_{\mathbf{xx}}^*(\mathbf{f} + \mathbf{L}\mathbf{w})\,\Delta t^2]$$

Functions of $\mathbf{x}(t)$ equal their own expectations, and $E[\mathbf{w}(t)] = \mathbf{0}$. Dividing by Δt, and replacing the third term by its trace, the time derivative is

$$\frac{dV^*}{dt} = V_t^* + V_{\mathbf{x}}^*\mathbf{f} + \tfrac{1}{2}\,\mathrm{Tr}\{E[(\mathbf{f} + \mathbf{L}\mathbf{w})^T V_{\mathbf{xx}}^*(\mathbf{f} + \mathbf{L}\mathbf{w})]\,\Delta t\}$$

$$= V_t^* + V_{\mathbf{x}}^*\mathbf{f} + \tfrac{1}{2}\,\mathrm{Tr}\{E[\,V_{\mathbf{xx}}^*(\mathbf{f} + \mathbf{L}\mathbf{w})(\mathbf{f} + \mathbf{L}\mathbf{w})^T\,]\,\Delta t\}$$

ADRL

ETH Zürich

Stochastic HJB

f, w uncorrelated

$$\frac{dV^*}{dt} = V_t^* + V_{\mathbf{x}}^* \mathbf{f} + \tfrac{1}{2} \lim_{\Delta t \to 0} \mathrm{Tr}\{V_{\mathbf{xx}}^*[E(\mathbf{f}\mathbf{f}^T)\,\Delta t + \mathbf{L}E(\mathbf{w}\mathbf{w}^T)\mathbf{L}^T\,\Delta t]\}$$

$$= V_t^* + V_{\mathbf{x}}^* \mathbf{f} + \tfrac{1}{2}\mathrm{Tr}(V_{\mathbf{xx}}^* \mathbf{L}\mathbf{W}\mathbf{L}^T)$$

plug in and rearrange

$$V_t^*(t) = -\min_{\mathbf{u}}\{\mathscr{L}[(\mathbf{x}^*(t), \mathbf{u}(t), t] + V_{\mathbf{x}}^* \mathbf{f}[\mathbf{x}^*(t), \mathbf{u}(t), t] + \tfrac{1}{2}\mathrm{Tr}[V_{\mathbf{xx}}^* \mathbf{L}(t)\mathbf{W}(t)\mathbf{L}^T(t)]\}$$

Terminal condition:

starting value for evaluation of $V^*(t)$ is $E\{\phi[\mathbf{x}(t_f), t_f]\}$, which is $\phi[\mathbf{x}(t_f), t_f]$ because $\mathbf{x}(t_f)$ can be measured without error.

)

nonlinear HJB

$$-\partial_t V_t = \min_{\mathbf{u}} \left(r_t + (\nabla_{\mathbf{x}} V_t)^T \mathbf{F}_t + \frac{1}{2} trace\left((\nabla_{\mathbf{xx}} V_t) \mathbf{G}_t \Sigma_\varepsilon \mathbf{G}_t^T \right) \right)$$

$$\mathbf{F}_t = \mathbf{f}(\mathbf{x}_t, t) + \mathbf{G}(\mathbf{x}_t)\mathbf{u}_t$$

1 3 gradient of RHS = 0 yields

$$\mathbf{u}(\mathbf{x}_t) = \mathbf{u}_t = -\mathbf{R}^{-1} \mathbf{G}_t^T (\nabla_{x_t} V_t)$$

4 substitute opt. control back into HJB ⇒

$$-\partial_t V_t = q_t + (\nabla_{\mathbf{x}} V_t)^T \mathbf{f}_t - \frac{1}{2} (\nabla_{\mathbf{x}} V_t)^T \mathbf{G}_t \mathbf{R}^{-1} \mathbf{G}_t^T (\nabla_{\mathbf{x}} V_t) + \frac{1}{2} trace\left((\nabla_{\mathbf{xx}} V_t) \mathbf{G}_t \Sigma_\varepsilon \mathbf{G}_t^T \right)$$

Nonlinear PDE!

A D R L

ETH Zürich

log transform

$$-\partial_t V_t = q_t + (\nabla_{\mathbf{x}} V_t)^T \mathbf{f}_t - \frac{1}{2}(\nabla_{\mathbf{x}} V_t)^T \mathbf{G}_t \mathbf{R}^{-1} \mathbf{G}_t^T (\nabla_{\mathbf{x}} V_t) + \frac{1}{2} trace\left((\nabla_{\mathbf{xx}} V_t)\mathbf{G}_t \Sigma_{\varepsilon} \mathbf{G}_t^T\right)$$

Nonlinear PDE!

$$\boxed{V_t = -\lambda \log \Psi_t}$$

$$\Rightarrow$$

$$\partial_t V_t = -\lambda \frac{1}{\Psi_t} \partial_t \Psi_t,$$

$$\nabla_{\mathbf{x}} V_t = -\lambda \frac{1}{\Psi_t} \nabla_{\mathbf{x}} \Psi_t,$$

$$\nabla_{\mathbf{xx}} V_t = \lambda \frac{1}{\Psi_t^2} \nabla_{\mathbf{x}} \Psi_t \, \nabla_{\mathbf{x}} \Psi_t^T - \lambda \frac{1}{\Psi_t} \nabla_{\mathbf{xx}} \Psi_t$$

$$\frac{\lambda}{\Psi_t} \partial_t \Psi_t = q_t - \frac{\lambda}{\Psi_t} (\nabla_{\mathbf{x}} \Psi_t)^T \mathbf{f}_t - \underline{\frac{\lambda^2}{2\Psi_t^2} (\nabla_{\mathbf{x}} \Psi_t)^T \mathbf{G}_t \mathbf{R}^{-1} \mathbf{G}_t^T (\nabla_{\mathbf{x}} \Psi_t)} + \frac{1}{2} trace(\Gamma)$$

$$\Gamma = \left(\lambda \frac{1}{\Psi_t^2} \nabla_{\mathbf{x}} \Psi_t \, \nabla_{\mathbf{x}} \Psi_t^T - \lambda \frac{1}{\Psi_t} \nabla_{\mathbf{xx}} \Psi_t\right) \mathbf{G}_t \Sigma_{\varepsilon} \mathbf{G}_t^T$$

ADRL

ETH Zürich

structure of control cost linked to noise

$$\frac{\lambda}{\Psi_t}\partial_t\Psi_t = q_t - \frac{\lambda}{\Psi_t}(\nabla_{\mathbf{x}}\Psi_t)^T\mathbf{f}_t - \underline{\frac{\lambda^2}{2\Psi_t^2}(\nabla_{\mathbf{x}}\Psi_t)^T\mathbf{G}_t\mathbf{R}^{-1}\mathbf{G}_t^T(\nabla_{\mathbf{x}}\Psi_t)} + \frac{1}{2}trace(\Gamma)$$

$$\Gamma = \left(\lambda\frac{1}{\Psi_t^2}\nabla_{\mathbf{x}}\Psi_t\,\nabla_{\mathbf{x}}\Psi_t^T - \lambda\frac{1}{\Psi_t}\nabla_{\mathbf{xx}}\Psi_t\right)\mathbf{G}_t\Sigma_\varepsilon\mathbf{G}_t^T$$

$$trace(\Gamma) = \underline{\lambda\frac{1}{\Psi^2}trace\left(\nabla_{\mathbf{x}}\Psi_t^T\mathbf{G}_t\Sigma_\varepsilon\mathbf{G}_t\nabla_{\mathbf{x}}\Psi_t\right)} - \lambda\frac{1}{\Psi_t}trace\left(\nabla_{\mathbf{xx}}\Psi_t\mathbf{G}_t\Sigma_\varepsilon\mathbf{G}_t^T\right)$$

$$\boxed{\lambda\mathbf{R}^{-1} = \Sigma_\varepsilon}$$

$$\lambda\mathbf{G}_t\mathbf{R}^{-1}\mathbf{G}_t^T = \mathbf{G}_t\Sigma_\varepsilon\mathbf{G}_t^T = \Sigma(\mathbf{x}_t) = \Sigma_t$$

$$-\partial_t\Psi_t = -\frac{1}{\lambda}q_t\Psi_t + \mathbf{f}_t^T(\nabla_{\mathbf{x}}\Psi_t) + \frac{1}{2}trace\left((\nabla_{\mathbf{xx}}\Psi_t)\mathbf{G}_t\Sigma_\varepsilon\mathbf{G}_t^T\right)$$

linear!

linear HJB

$$-\partial_t \Psi_t = -\frac{1}{\lambda} q_t \Psi_t + \mathbf{f}_t^T (\nabla_{\mathbf{x}} \Psi_t) + \frac{1}{2} trace\left((\nabla_{\mathbf{xx}} \Psi_t) \mathbf{G}_t \Sigma_\varepsilon \mathbf{G}_t^T \right)$$

linear, but still no analytic solution for arbitrary q(x,t)

solve backward terminal condition : $\Psi_{t_N} = \exp\left(-\frac{1}{\lambda}\phi_{t_N}\right)$

Feynman-Kac Theorem: Can write solution of PDE as Expectation over stochastic forward dynamics

$$\Psi_{t_i} = E_{\tau_i}\left(\Psi_{t_N} e^{-\int_{t_i}^{t_N} \frac{1}{\lambda} q_t dt} \right) = E_{\tau_i}\left[\exp\left(-\frac{1}{\lambda}\phi_{t_N} - \frac{1}{\lambda}\int_{t_i}^{t_N} q_t \, dt \right) \right]$$

forward! ... but stochastic

Remember the forward search in the discrete state, discrete time problem (Lect. 2)

Expectations over paths

$$\Psi_{t_i} = E_{\tau_i}\left(\Psi_{t_N} e^{-\int_{t_i}^{t_N} \frac{1}{\lambda} q_t dt}\right) = E_{\tau_i}\left[\exp\left(-\frac{1}{\lambda}\phi_{t_N} - \frac{1}{\lambda}\int_{t_i}^{t_N} q_t \, dt\right)\right]$$

forward!

... but stochastic

$$\int p(\tau) \exp\left(-\frac{1}{\lambda}\phi - \frac{1}{\lambda}\int q dt\right) d\tau$$

$$\tau = x(t \ldots t_N) \sim p(x, u)$$

an instance of a random path segment (a random 'number', but in spaces of functions)

$$E[X] = \int x p(x) dx$$

$$x = f(t)$$

Continuous time, x is function of time

Major difficulty: Definition of stochastic processes in continuous time!

		states	
		discrete	continuous
time	discrete	$\underset{\int \to \Sigma}{\overset{dx \to 0}{\rightleftarrows}}$	$\int \to \Sigma \uparrow \downarrow dt \to 0$
	continuous		Lots of gnarly math!

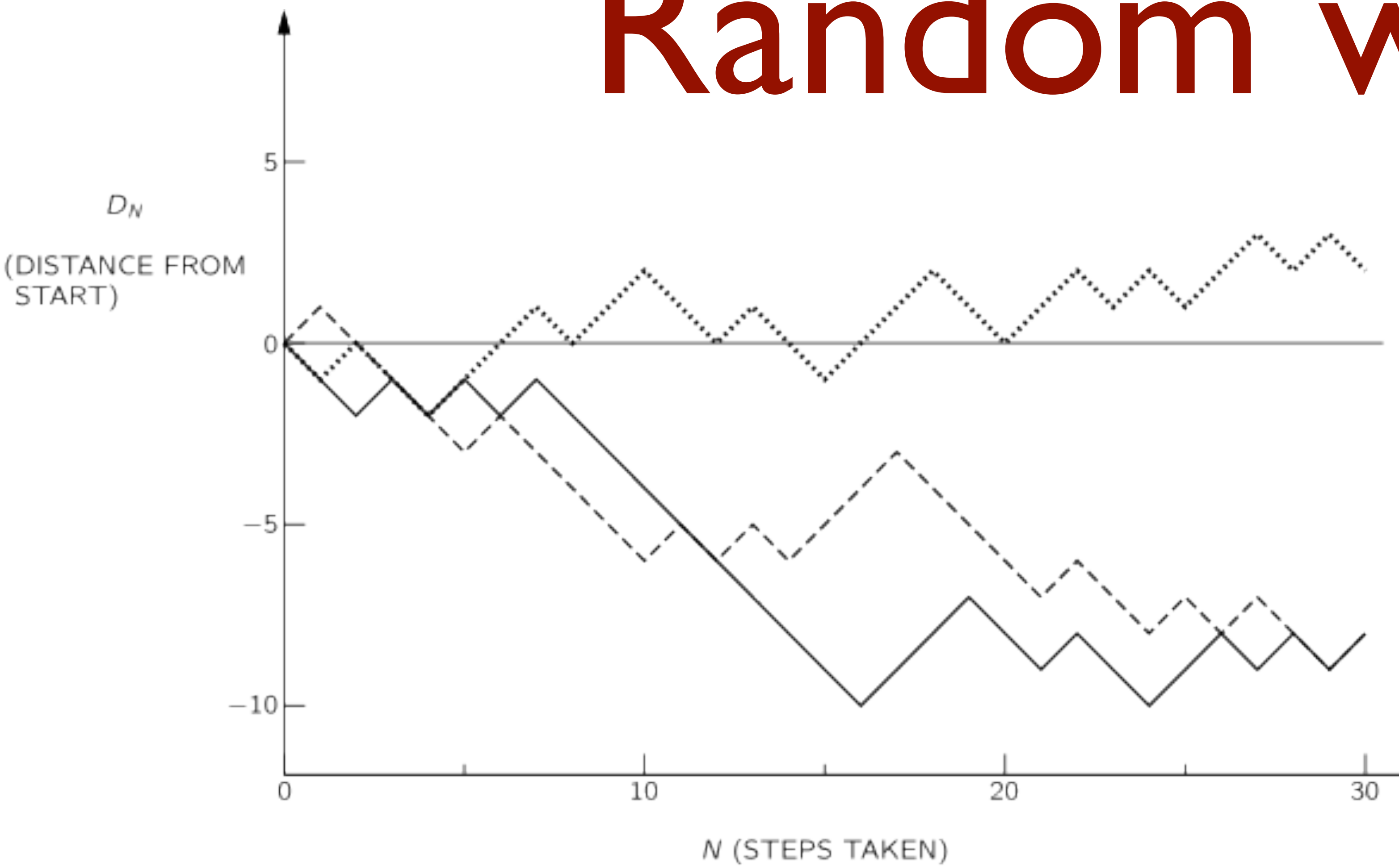
A D R L

ETH Zürich

Continuous Random Processes

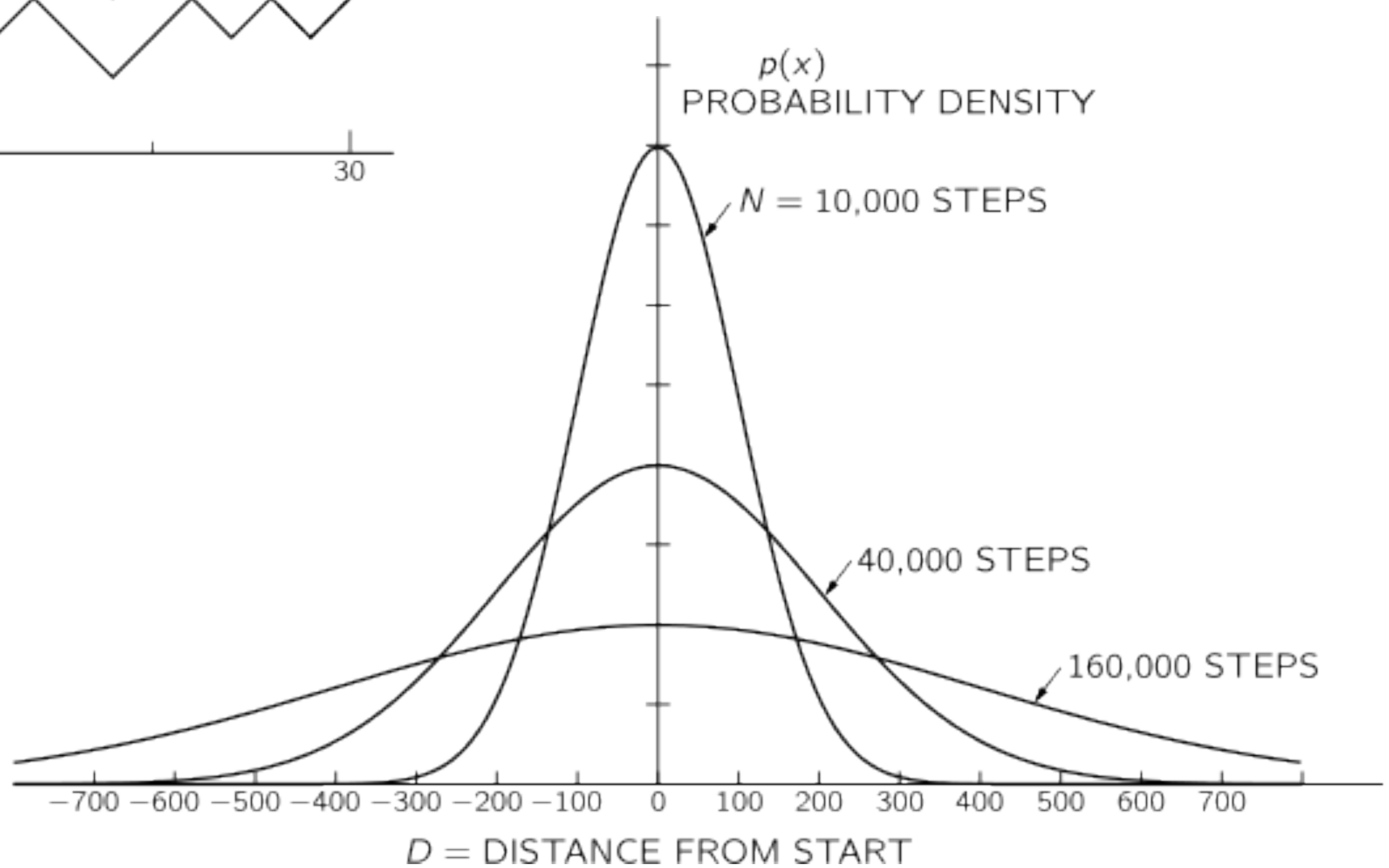
$$d\mathbf{x} = \mathbf{f}\left(\mathbf{x}, \mathbf{u}\right) dt + F\left(\mathbf{x}, \mathbf{u}\right) d\boldsymbol{\omega}$$

Random walks



‘paths diffuse over time’

‘density’



Continuous decision processes

Take random walk and take limits

$$dx \to 0$$ probability densities

$$dt \to 0$$ probability flow

for all times $\int p(x)dx = 1$ conservation law!

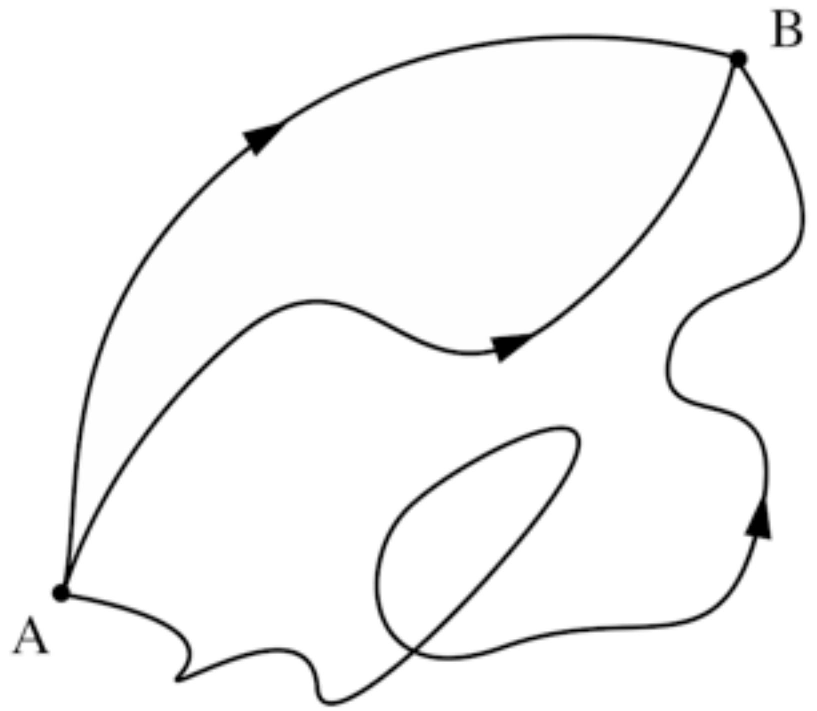
Conserved flow?
You know how to do that!

A D R L

ETH Zürich

Comparison to graphs

can think of all possibilities of a random walk as graph

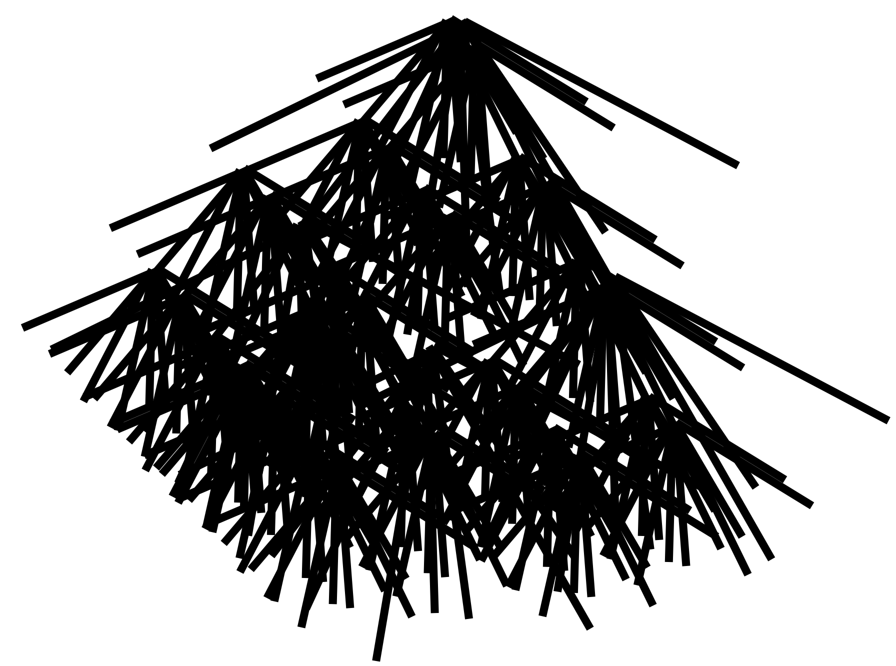


There are several ways to end up in a certain state, each path has an associated probability

When does 'branching' occur?

Idea: do discrete time and take limit

$$dt \rightarrow 0$$



A D R L

ETH Zürich

Probabilistic Dynamics

Discrete time:

Markov chains — Master Equation

Continuous time:

Jumps: Continuous-time Markov chain

Smooth: Markov Process

Fokker-Planck

cf. (Heat) Diffusion

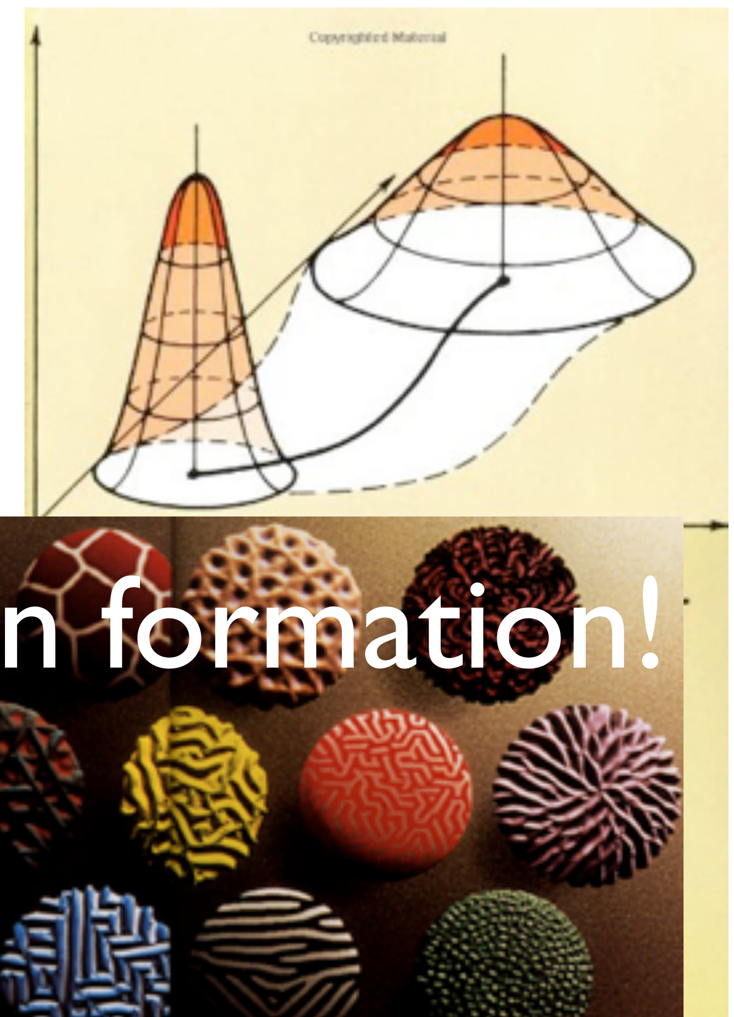
Fokker-Planck Equation

(the most interesting equation in the world?)

PDE for time evolution of probability distribution

$$\frac{\partial}{\partial t}p(x,t) = -\frac{\partial}{\partial x}[\mu(x,t)p(x,t)] + \frac{\partial^2}{\partial x^2}[D(x,t)p(x,t)]$$

Drift Diffusion



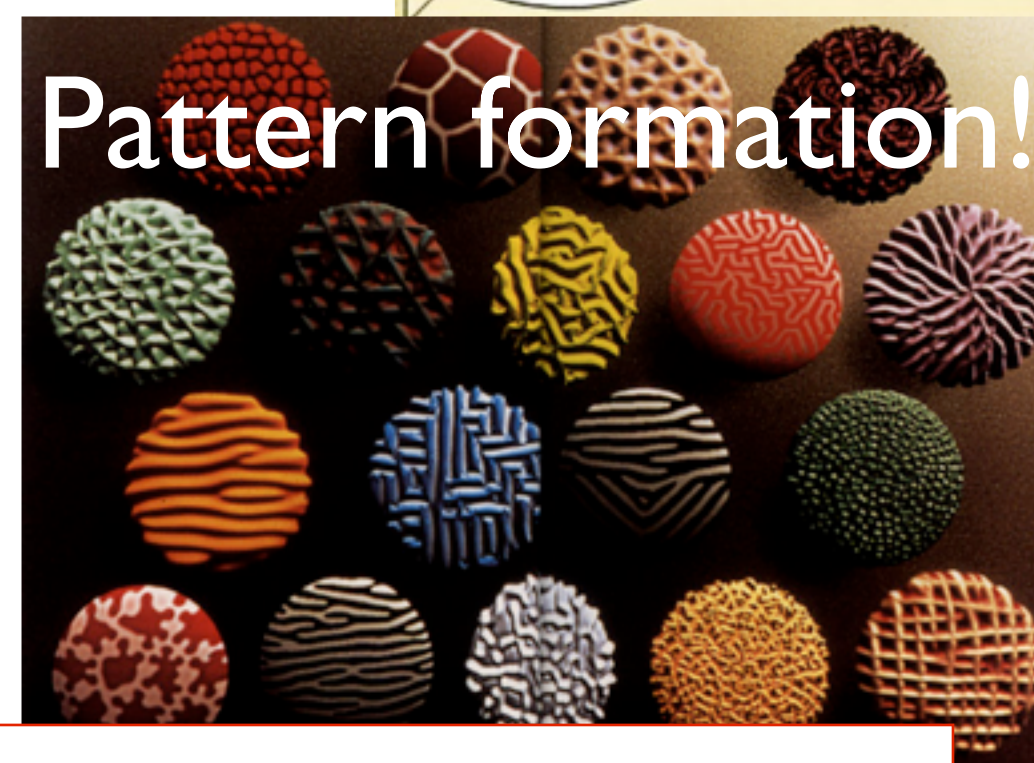
conservation law!

$$\int p(x)dx = 1$$

$dX_t = dW_t.$ brownian motion, no drift

$$\frac{\partial p(x,t)}{\partial x} = \frac{1}{2}\frac{\partial^2 p(x,t)}{\partial x^2}$$

$$\Rightarrow p(x,t) = \frac{1}{\sqrt{2\pi t}}e^{-\frac{x^2}{2t}}$$

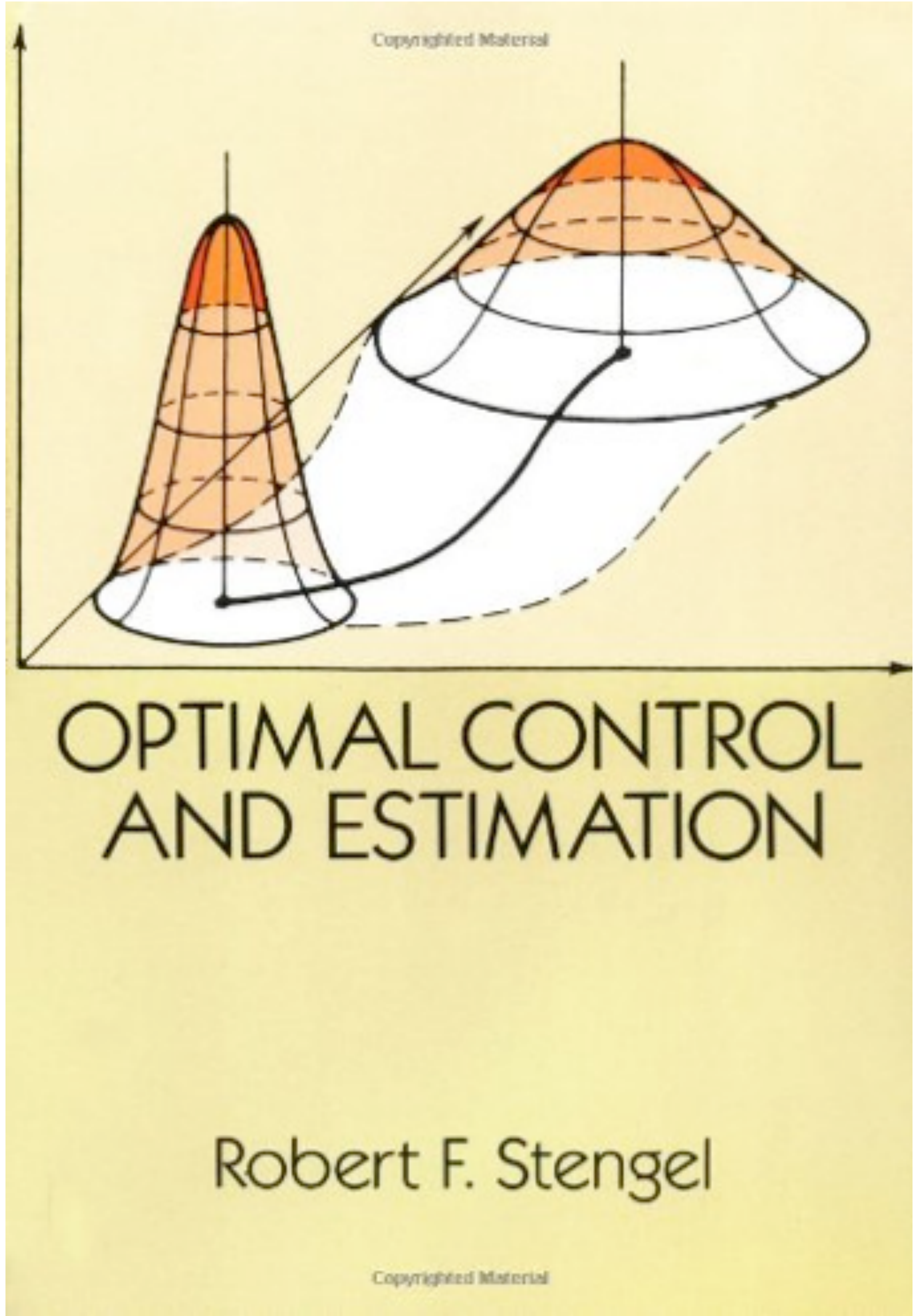


cf. Fluid Dynamics
Heat and Charge diffusion
cf. Particle filters

Finance, Biology, Chemistry, Physics, Sociology, Anthropology, Control & Machine Learning

Stochastic Control

'Controlled Diffusion'
Controlled Brownian Motion



Example

High dimensional continuous state actions spaces with stochastic dynamics

Optimal(?) control in fluids

Approach: Computational Fluid dynamics & Evolutionary Algorithm


$C =$ 'don't eat me!'
$$\frac{\partial C}{\partial \theta} = \frac{\partial\text{'don't eat me!'}}{\partial\text{'how to flap???'}}$$
Photograph by Joel Sartore
National Geographic Magazine Online
ngm.com/wildcamgrizzlies, June 2006
© 2006 National Geographic Society. All rights reserved.

$$\rho\left(\frac{\partial \mathbf{v}}{\partial t} + \mathbf{v}\cdot\nabla\mathbf{v}\right) = -\nabla p + \nabla\cdot\mathbf{T} + \mathbf{f}$$

)

$$V_t = -\lambda \log \Psi_t$$

$$\Psi_{t_i} = E\tau_i\left(\Psi_{t_N} e^{-\int_{t_i}^{t_N} \frac{1}{\lambda} q_t dt}\right) = E\tau_i\left[\exp\left(-\frac{1}{\lambda}\phi_{t_N} - \frac{1}{\lambda}\int_{t_i}^{t_N} q_t \, dt\right)\right]$$

forward! ... but stochastic

$$\int p(\tau) \exp\left(-\frac{1}{\lambda}\phi - \frac{1}{\lambda}\int q dt\right) d\tau$$

discretize

$$\tau_i = (\mathbf{x}_{t_i}, \mathbf{x}_{t_{i+1}}, \ldots.., \mathbf{x}_{t_N}) \qquad d\tau_i = (d\mathbf{x}_{t_i}, \ldots.., d\mathbf{x}_{t_N})$$

path starting at t_i to end of episode

'how much does it cost'?

$$\Psi_{t_i} = \lim_{dt \to 0} \int p(\tau_i | \mathbf{x}_i) \exp\left[-\frac{1}{\lambda}\left(\phi_{t_N} + \sum_{j=i}^{N-1} q_{t_j} dt\right)\right] d\tau_i$$

'where do i end up next?'

integrate ('sum') over all possible paths: 'path integral'

$$p(\tau_i | \mathbf{x}_i)???$$

ADRL

ETH Zürich

$$
\begin{aligned}
p(\tau_i|\mathbf{x}_{t_i}) &= p(\tau_{i+1}|\mathbf{x}_{t_i}) \\
&= p(\mathbf{x}_{t_N}, \ldots\ldots, \mathbf{x}_{t_{i+1}}|\mathbf{x}_{t_i}) \\
&= \Pi_{j=i}^{N-1} p(\mathbf{x}_{t_{j+1}}|\mathbf{x}_{t_j}),
\end{aligned}
$$

Gaussian noise leads to

$$
p\left(\mathbf{x}_{t_{j+1}}^{(c)}|\mathbf{x}_{t_j}\right) = \frac{1}{\left((2\pi)^l \cdot |\Sigma_{t_j}|\right)^{1/2}} \exp\left(-\frac{1}{2}\left\|\mathbf{x}_{t_{j+1}}^{(c)} - \mathbf{x}_{t_j}^{(c)} - \mathbf{f}_{t_j}^{(c)} dt\right\|^2_{\Sigma_{t_j}^{-1}}\right)
$$

'deviation from deterministic dynamics'

$$
\boxed{
\begin{aligned}
\Psi_{t_i} &= \lim_{dt\to 0} \int \exp\left(-\frac{1}{\lambda} S(\tau_i) - \log D(\tau_i)\right) d\tau_i^{(c)} \\
&= \lim_{dt\to 0} \int \exp\left(-\frac{1}{\lambda} Z(\tau_i)\right) d\tau_i^{(c)},
\end{aligned}
}
$$

'effect of noise variance'

$$
S(\tau_i) = \phi_{t_N} + \sum_{j=i}^{N-1} q_{t_j} dt + \boxed{\frac{1}{2} \sum_{j=i}^{N-1} \left\| \frac{\mathbf{x}_{t_{j+1}}^{(c)} - \mathbf{x}_{t_j}^{(c)}}{dt} - \mathbf{f}_{t_j}^{(c)} \right\|^2_{\mathbf{H}_{t_j}^{-1}}} dt
$$

$$
D(\tau_i) = \Pi_{j=i}^{N-1} \left((2\pi)^{l/2} |\Sigma_{t_j}|^{1/2}\right)
$$

'normalization with noise variance'

A D R L

ETH Zürich

Illustration

[Psi at t_i]

$$V_t = -\lambda \log \Psi_t$$

$$\partial_t V_t = -\lambda \frac{1}{\Psi_t} \partial_t \Psi_t,$$

$$\nabla_{\mathbf{x}} V_t = -\lambda \frac{1}{\Psi_t} \nabla_{\mathbf{x}} \Psi_t,$$

$$\nabla_{\mathbf{xx}} V_t = \lambda \frac{1}{\Psi_t^2} \nabla_{\mathbf{x}} \Psi_t \nabla_{\mathbf{x}} \Psi_t^T - \lambda \frac{1}{\Psi_t} \nabla_{\mathbf{xx}} \Psi_t$$

$$\mathbf{u}_{t_i} = -\mathbf{R}^{-1} \mathbf{G}_{t_i}^T (\nabla_{x_{t_i}} V_{t_i})$$

$$\Rightarrow \quad \mathbf{u}_{t_i} = \lambda \mathbf{R}^{-1} \mathbf{G}_{t_i} \frac{\nabla_{\mathbf{x}_{t_i}} \Psi_{t_i}}{\Psi_{t_i}}$$

Plug in Ψ

$$\mathbf{u}_{t_i} = \lim_{dt \to 0} \left(\lambda \mathbf{R}^{-1} \mathbf{G}_{t_i}^T \frac{\nabla_{\mathbf{x}_{t_i}^{(c)}} \left(\int e^{-\frac{1}{\lambda} \tilde{S}(\tau_i)} d\tau_i^{(c)} \right)}{\int e^{-\frac{1}{\lambda} \tilde{S}(\tau_i)} d\tau_i^{(c)}} \right)$$

$$\boxed{\mathbf{u}_{t_i} = \int P(\tau_i) \mathbf{u}_L(\tau_i) d\tau_i^{(c)}} \qquad \mathbf{u}_L(\tau_i) = -\mathbf{R}^{-1} \mathbf{G}_{t_i}^{(c)T} \lim_{dt \to 0} \left(\nabla_{\mathbf{x}_{t_i}^{(c)}} \tilde{S}(\tau_i) \right)$$

$$\boxed{P(\tau_i) = \frac{e^{-\frac{1}{\lambda} \tilde{S}(\tau_i)}}{\int e^{-\frac{1}{\lambda} \tilde{S}(\tau_i)} d\tau_i}} \qquad \boxed{\mathbf{u}_L(\tau_i) = \mathbf{R}^{-1} \mathbf{G}_{t_i}^{(c)T} \left(\mathbf{G}_{t_i}^{(c)} \mathbf{R}^{-1} \mathbf{G}_{t_i}^{(c)T} \right)^{-1} \mathbf{G}_{t_i}^{(c)} \varepsilon_{t_i}}$$

'project noise in range space of control gain'
- weighted with control cost R

Example: Naive sampling

$$M(\theta) \cdot \ddot{\theta} + C(\theta, \dot{\theta}) = \tau$$

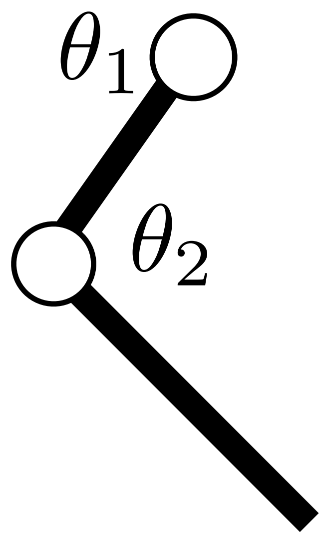
$$\ddot{\theta} = M(\theta)^{-1} \cdot \left(-C(\theta, \dot{\theta}) + \tau\right)$$

$$M(\theta) = \begin{pmatrix} d_1 + 2d_2\cos(\theta_2) & d_3 + d_2\cos(\theta_2) \\ d_3 + d_2\cos(\theta_2) & d_3 \end{pmatrix} \quad (37)$$

$$C(\dot{\theta}, \theta) = \begin{pmatrix} -\dot{\theta}_2\left(2\dot{\theta}_1 + \dot{\theta}_2\right) \\ \dot{\theta}_1^2 \end{pmatrix} d_2 \sin(\theta_2) \quad (38)$$

$$d_1 = I_1 + I_2 + m_2 l_1^2, \quad d_2 = m_2 l_1 s_2, \quad d_3 = I_2 \quad (39)$$

Symbol	Value	Unit
m_1	1.4	Kg
m_2	1	Kg
s_1	0.11	m
s_2	0.16	m
I_1	0.3	Kg m^2
I_2	0.33	Kg m^2
l_1	0.025	m
l_2	0.045	m



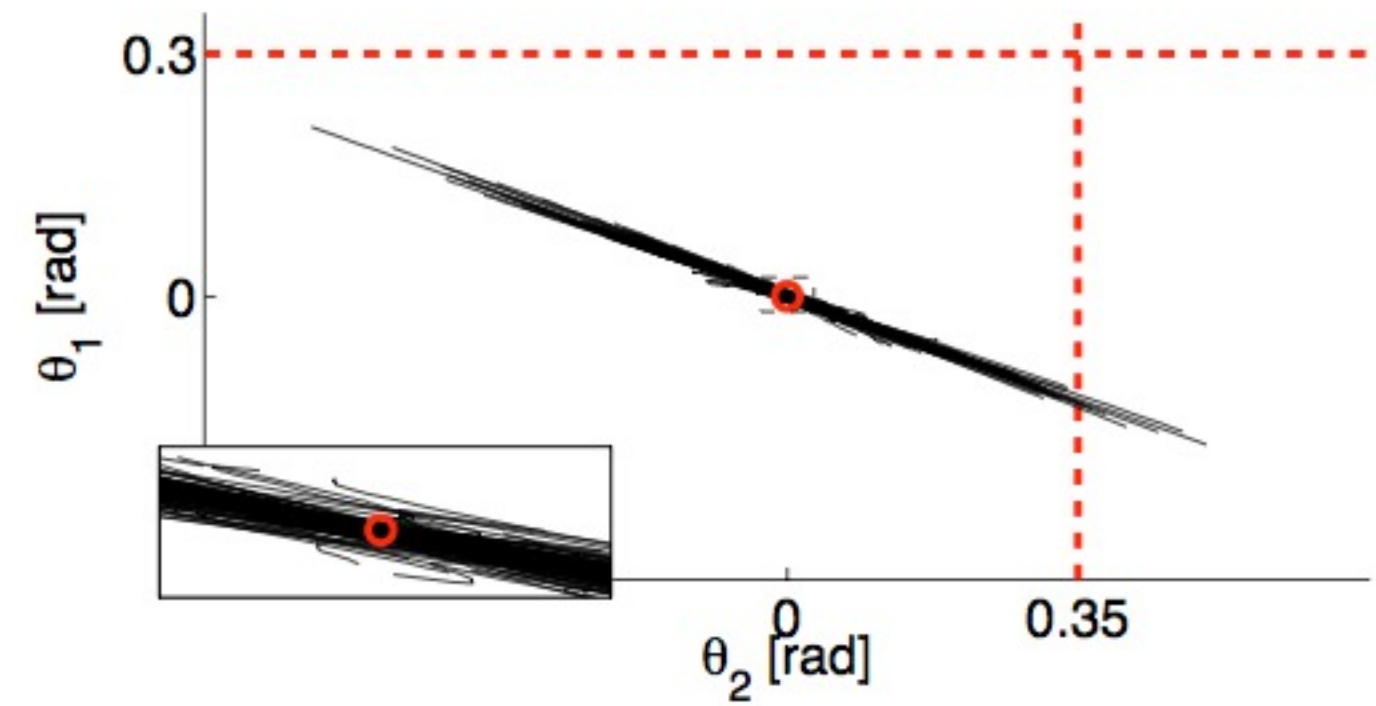
$$\dot{x} = \Phi(x) + G(x) \cdot \tau$$

$$x = \begin{pmatrix} \theta_1 & \theta_2 & \dot{\theta}_1 & \dot{\theta}_2 \end{pmatrix}^T \quad \tau = \begin{pmatrix} \tau_1 & \tau_2 \end{pmatrix}.$$

30)

$$\Phi(x) = \begin{pmatrix} \dot{\theta}_1 \\ \dot{\theta}_2 \\ -M(\theta)^{-1} \cdot C(\theta, \dot{\theta}) \end{pmatrix}, \quad G(x) = \begin{pmatrix} O_{2\times 2} \\ M(\theta)^{-1} \end{pmatrix}$$

Path integral SOC requires sampling of passive dynamics with gaussian mean-free noise



A D R L

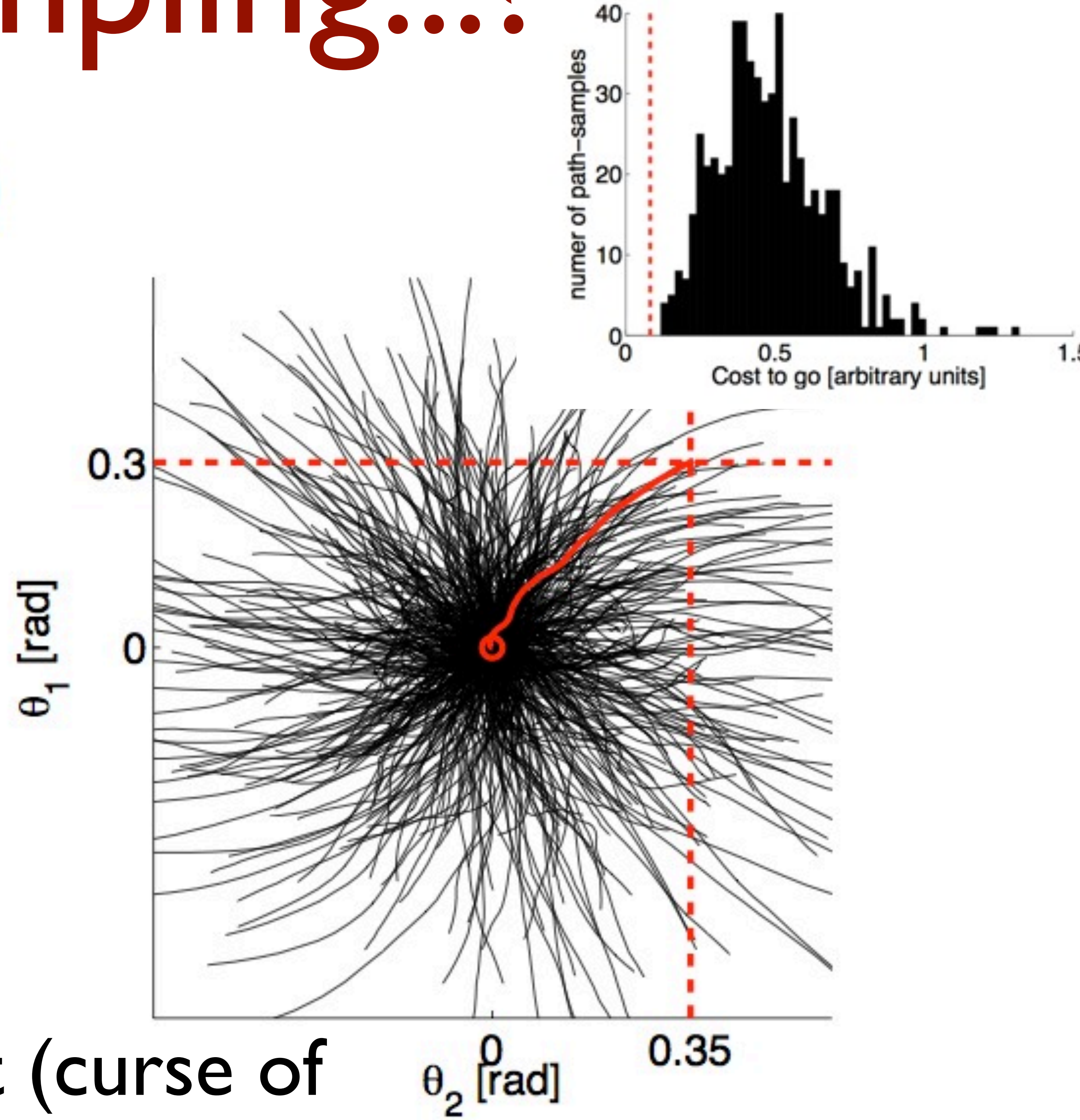
Bucl

Improved sampling...?

$$\tau_u^i = M(\theta) \cdot (\alpha_i + \epsilon_i) + C(\theta, \dot{\theta})$$

$$M(\theta) \cdot \ddot{\theta} + C(\theta, \dot{\theta}) = \tau_u$$

Sample in acceleration space, use inverse dynamics controllers to find torques:



... still not very efficient (curse of dimensionality still strikes, needle in a haystack!)

From Model based to Model-free PISOC

$$\mathbf{u}_{t_i} = \int P(\tau_i)\,\mathbf{u}_L(\tau_i)\,d\tau_i^{(c)}$$

$$P(\tau_i) = \frac{e^{-\frac{1}{\lambda}\tilde{S}(\tau_i)}}{\int e^{-\frac{1}{\lambda}\tilde{S}(\tau_i)}\,d\tau_i}$$

$$\mathbf{u}_L(\tau_i) = \mathbf{R}^{-1}\mathbf{G}_{t_i}^{(c)T}\left(\mathbf{G}_{t_i}^{(c)}\mathbf{R}^{-1}\mathbf{G}_{t_i}^{(c)T}\right)^{-1}\mathbf{G}_{t_i}^{(c)}\boldsymbol{\varepsilon}_{t_i}$$

Optimal control need model: Input Gain matrix

Good sampling: need model, input gain matrix

Policy improvements with Path Integrals - PI2

1)Local sampling, iterative method

2)Use an 'intermediate system' with known input gain matrix: parametrized policies!

parameters

$$\mathbf{a}_{t_i} = \mathbf{g}_{t_i}^T(\theta + \varepsilon_{t_i})$$

basis functions

noise

Path integral SOC with parameterized policy

$$\mathbf{a}_{t_i} = \mathbf{g}_{t_i}^T(\theta + \varepsilon_{t_i})$$

$$\boxed{\mathbf{u}_{t_i} = \int P(\tau_i)\,\mathbf{u}_L(\tau_i)\,d\tau_i^{(c)}}$$

$$P(\tau_i) = \frac{e^{-\frac{1}{\lambda}\tilde{S}(\tau_i)}}{\int e^{-\frac{1}{\lambda}\tilde{S}(\tau_i)}d\tau_i}, \quad \mathbf{u}_L(\tau_i) = \frac{\mathbf{R}^{-1}\mathbf{g}_{t_i}^{(c)}\mathbf{g}_{t_i}^{(c)T}}{\mathbf{g}_{t_i}^{(c)T}\mathbf{R}^{-1}\mathbf{g}_{t_i}^{(c)}}\varepsilon_{t_i},$$

A D R L

ETH Zürich

Iterative Path integrals with Parametrized policies

$$\mathbf{a}_{t_i} = \mathbf{g}_{t_i}^T(\theta + \varepsilon_{t_i})$$

$$\boxed{\mathbf{u}_{t_i} = \int P(\tau_i)\,\mathbf{u}_L(\tau_i)\,d\tau_i^{(c)}},$$

$$P(\tau_i) = \frac{e^{-\frac{1}{\lambda}\tilde{S}(\tau_i)}}{\int e^{-\frac{1}{\lambda}\tilde{S}(\tau_i)}d\tau_i}, \quad \mathbf{u}_L(\tau_i) = \frac{\mathbf{R}^{-1}\mathbf{g}_{t_i}^{(c)}\mathbf{g}_{t_i}^{(c)T}}{\mathbf{g}_{t_i}^{(c)T}\mathbf{R}^{-1}\mathbf{g}_{t_i}^{(c)}}\varepsilon_{t_i}$$

$$\tilde{S}(\tau_i) = \phi_{t_N} + \sum_{j=i}^{N-1} q_{t_j} + \frac{1}{2}\sum_{j=i}^{N-1} \varepsilon_{t_j}^T \mathbf{M}_{t_j}^T \mathbf{R} \mathbf{M}_{t_j} \varepsilon_{t_j}$$

$$\mathbf{M}_{t_j} = \frac{\mathbf{R}^{-1}\mathbf{g}_{t_j}\mathbf{g}_{t_j}^T}{\mathbf{g}_{t_j}^T\mathbf{R}^{-1}\mathbf{g}_{t_j}}$$

$$\begin{aligned}
\theta_{t_i}^{(new)} &= \int P(\tau_i)\frac{\mathbf{R}^{-1}\mathbf{g}_{t_i}\mathbf{g}_{t_i}{}^T(\theta + \varepsilon_{t_i})}{\mathbf{g}_{t_i}{}^T\mathbf{R}^{-1}\mathbf{g}_{t_i}}d\tau_i \\
&= \int P(\tau_i)\frac{\mathbf{R}^{-1}\mathbf{g}_{t_i}\mathbf{g}_{t_i}{}^T\varepsilon_{t_i}}{\mathbf{g}_{t_i}{}^T\mathbf{R}^{-1}\mathbf{g}_{t_i}}d\tau_i + \frac{\mathbf{R}^{-1}\mathbf{g}_{t_i}\mathbf{g}_{t_i}{}^T\theta}{\mathbf{g}_{t_i}{}^T\mathbf{R}^{-1}\mathbf{g}_{t_i}} \\
&= \delta\theta_{t_i} + \frac{\mathbf{R}^{-1}\mathbf{g}_{t_i}\mathbf{g}_{t_i}{}^T}{trace\left(\mathbf{R}^{-1}\mathbf{g}_{t_i}\mathbf{g}_{t_i}{}^T\right)}\theta \\
&= \delta\theta_{t_i} + \mathbf{M}_{t_i}\theta.
\end{aligned}$$

Compare to DDP

A D R L

ETH zürich

PI^2

Update step

$$
\begin{aligned}
P(\tau_i) &= \frac{e^{-\frac{1}{\lambda}S(\tau_i)}}{\int e^{-\frac{1}{\lambda}S(\tau_i)}d\tau_i}, & (35)\\
S(\tau_i) &= \phi_{t_N} + \sum_{j=i}^{N-1} q_{t_j}dt + \frac{1}{2}\sum_{j=i}^{N-1}(\theta + \mathbf{M}_{t_j}\varepsilon_{t_j})^T\mathbf{R}(\theta + \mathbf{M}_{t_j}\varepsilon_{t_j})dt, & (36)\\
\delta\theta_{t_i} &= \int P(\tau_i)\,\mathbf{M}_{t_i}\varepsilon_{t_i}d\tau_i, & (37)\\
[\delta\theta]_j &= \frac{\sum_{i=0}^{N-1}(N-i)\,w_{j,t_i}\,[\delta\theta_{t_i}]_j}{\sum_{i=0}^{N-1}w_{j,t_i}(N-i)}, & (38)\\
\theta^{(new)} &= \theta^{(old)} + \delta\theta.
\end{aligned}
$$

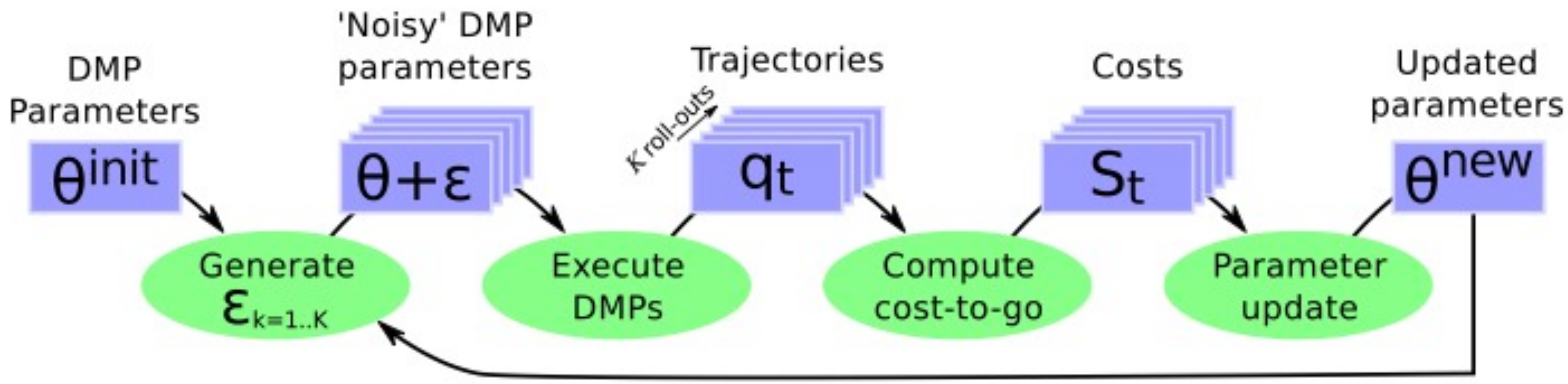


Fig. 2. Overview of the PI^2 algorithm.

PI^2

Policy Improvement with Path Integrals

- **Given**:
  - An immediate cost function $r_t = q_t + \theta_t^T \mathbf{R} \theta_t$ (cf. 1)
  - A terminal cost term ϕ_{t_N} (cf. 1)
  - A stochastic parameterized policy $\mathbf{a}_t = \mathbf{g}_t^T(\theta + \varepsilon_t)$ (cf. 25)
  - The basis function $\mathbf{g}_{t_i}$ from the system dynamics (cf. 3 and Section 2.5.1)
  - The variance Σ_ε of the mean-zero noise ε_t
  - The initial parameter vector θ

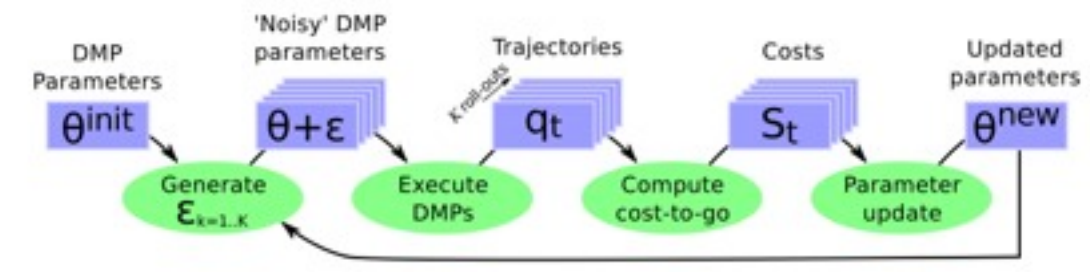


Fig. 2. Overview of the PI^2 algorithm.

- **Repeat** until convergence of the trajectory cost R:
  - Create K roll-outs of the system from the same start state $\mathbf{x}_0$ using stochstic parameters $\theta + \varepsilon_t$ at every time step
  - **For** $k = 1...K$, compute:
    * $P(\tau_{i,k}) = \frac{e^{-\frac{1}{\lambda}S(\tau_{i,k})}}{\sum_{k=1}^{K}[e^{-\frac{1}{\lambda}S(\tau_{i,k})}]}$
    * $S(\tau_{i,k}) = \phi_{t_N,k} + \sum_{j=i}^{N-1} q_{t_j,k} + \frac{1}{2}\sum_{j=i+1}^{N-1}(\theta + \mathbf{M}_{t_j,k}\varepsilon_{t_j,k})^T \mathbf{R}(\theta + \mathbf{M}_{t_j,k}\varepsilon_{t_j,k})$
    * $\mathbf{M}_{t_j,k} = \frac{\mathbf{R}^{-1}\mathbf{g}_{t_j,k}\,\mathbf{g}_{t_j,k}^T}{\mathbf{g}_{t_j,k}^T \mathbf{R}^{-1}\mathbf{g}_{t_j,k}}$
  - **For** $i = 1...(N-1)$, compute:
    * $\delta\theta_{t_i} = \sum_{k=1}^{K}[P(\tau_{i,k})\,\mathbf{M}_{t_i,k}\,\varepsilon_{t_i,k}]$
  - Compute $[\delta\theta]_j = \frac{\sum_{i=0}^{N-1}(N-i)\, w_{j,t_i}\,[\delta\theta_{t_i}]_j}{\sum_{i=0}^{N-1} w_{j,t_i}(N-i)}$
  - Update $\theta \leftarrow \theta + \delta\theta$
  - Create one noiseless roll-out to check the trajectory cost $R = \phi_{t_N} + \sum_{i=0}^{N-1} r_{t_i}$. In case the noise cannot be turned off, that is, a stochastic system, multiple roll-outs need be averaged.

ETH zürich

Simplifications to PI2

$$S(\boldsymbol{\tau}_{i,k}) = \phi_{t_N,k} + \sum_{j=i}^{N-1} r_{t_j,k} + \cancel{\frac{1}{2}\sum_{j=i+1}^{N-1}} (\boldsymbol{\theta} + \mathbf{M}_{t_j,k}\boldsymbol{\epsilon}^{\boldsymbol{\theta}}{}_{t_j,k})^T \mathbf{R}(\boldsymbol{\theta} + \mathbf{M}_{t_j,k}\boldsymbol{\epsilon}^{\boldsymbol{\theta}}{}_{t_j,k}) \quad (7)$$

$$\mathbf{M}_{t_j,k} = \frac{\mathbf{R}^{-1}\mathbf{g}_{t_j}\ \mathbf{g}_{t_j}^T}{\mathbf{g}_{t_j}^T \mathbf{R}^{-1}\mathbf{g}_{t_j}} \quad (8)$$

$$P(\boldsymbol{\tau}_{i,k}) = \frac{e^{-\frac{1}{\lambda}S(\boldsymbol{\tau}_{i,k})}}{\sum_{l=1}^{K}[e^{-\frac{1}{\lambda}S(\boldsymbol{\tau}_{i,l})}]} \quad (9)$$

$$\delta\boldsymbol{\theta}_{t_i} \cancel{= \sum_{k=1}^{K} [P(\boldsymbol{\tau}_{i,k})\ \mathbf{M}_{t_i,k}}\ \boldsymbol{\epsilon}^{\boldsymbol{\theta}}{}_{t_i,k}] \quad (10)$$

$$[\delta\boldsymbol{\theta}]_j = \frac{\sum_{i=0}^{N-1}(N-i)\ w_{j,t_i}\ [\delta\boldsymbol{\theta}_{t_i}]_j}{\sum_{i=0}^{N-1} w_{j,t_i}(N-i)} \quad (11)$$

$$\boldsymbol{\theta} \leftarrow \boldsymbol{\theta} + \delta\boldsymbol{\theta} \quad (12)$$

Simplifications

M = 1

Only use total cost to go

$$P\left(\boldsymbol{\tau}_{0,k}\right) = \frac{e^{-\frac{1}{\lambda}J(\boldsymbol{\tau}_{0,k})}}{\sum_{l=1}^{K}[e^{-\frac{1}{\lambda}J(\boldsymbol{\tau}_{0,l})}]} \quad \text{Probability} \quad (21)$$

$$\delta g = \sum_{k=1}^{K}[P\left(\boldsymbol{\tau}_{0,k}\right)\ \epsilon^{g}{}_{k}] \quad \text{Weighted averaging} \quad (22)$$

$$g \leftarrow g + \delta g \quad \text{Update} \quad (23)$$

... a few more things

Elitism: Remember overall best few and use in update

Lambda: Use schedule to 'freeze' the system

Credits & Refs

Path Integral Based Stochastic Optimal Control for Rigid Body Dynamics
E. A. Theodorou, J. Buchli and S. Schaal

A Generalized Path Integral Control Approach to Reinforcement Learning.
Evangelos A. Theodorou, Jonas Buchli, Stefan Schaal

Learning Motion Primitive Goals for Robust Manipulation
Freek Stulp, Evangelos Theodorou, Mrinal Kalakrishnan, Peter Pastor, Ludovic Righetti, Stefan Schaal

Feynman Lectures on Physics

EOF L7