Design (and Implementation) of PDMP Monte Carlo Methods

BAMC 2022 Minisymposium

Nonreversible Processes: Analysis and Computations

Sam Power, 13 Apr 2022

Problem Setting

Problem Setting

- goal: generate samples from $\pi(x) = \exp\left(-U(x)\right)/Z$

Problem Setting

- goal: generate samples from $\pi(x) = \exp\left(-U(x)\right)/Z$
  - easy to evaluate U, Z not so much

Problem Setting

- goal: generate samples from $\pi(x) = \exp\left(-U(x)\right)/Z$
  - easy to evaluate U, Z not so much
    - if $\dim x \gg 1$, direct strategies unlikely to work

Problem Setting

- goal: generate samples from $\pi(x) = \exp\left(-U(x)\right)/Z$
  - easy to evaluate U, Z not so much
    - if $\dim x \gg 1$, direct strategies unlikely to work
- strategy: iterative method

Problem Setting

- goal: generate samples from $\pi(x) = \exp\left(-U(x)\right)/Z$
    - easy to evaluate U, Z not so much
        - if $\dim x \gg 1$, direct strategies unlikely to work
- strategy: iterative method
    - generate a sequence $\left(x_t\right)_{t \geqslant 0}$ such that $\mathrm{Law}\left(x_t\right) \overset{t \to \infty}{\to} \pi$

Problem Setting

- goal: generate samples from $\pi(x) = \exp\left(-U(x)\right)/Z$
  - easy to evaluate U, Z not so much
    - if $\dim x \gg 1$, direct strategies unlikely to work
- strategy: iterative method
  - generate a sequence $\left(x_t\right)_{t\geqslant 0}$ such that $\mathrm{Law}\left(x_t\right) \overset{t\to\infty}{\to} \pi$
    - for simplicity: x_t memoryless, stationary (“MCMC”)

Dynamical Systems for Sampling

(focus on *local* dynamics, in *continuous time* (makes life easier)**)**

Dynamical Systems for Sampling

(focus on *local* dynamics, in *continuous time* (makes life easier)**)**

- what can dynamics involve?

Dynamical Systems for Sampling

(focus on *local* dynamics, in *continuous time* (makes life easier)**)**

- what can dynamics involve?
    1. following a vector field

Dynamical Systems for Sampling

(focus on *local* dynamics, in *continuous time* (makes life easier)**)**

- what can dynamics involve?
  1. following a vector field
  2. diffusing locally according to some metric

Dynamical Systems for Sampling

(focus on *local* dynamics, in *continuous time* (makes life easier)**)**

- what can dynamics involve?
    1. following a vector field
    2. diffusing locally according to some metric
    3. ‘structured’ jumps

Dynamical Systems for Sampling

(focus on *local* dynamics, in *continuous time* (makes life easier)**)**

- what can dynamics involve?
    1. following a vector field
    2. diffusing locally according to some metric
    3. ‘structured’ jumps
- have some fun: hybrid strategies

Dynamical Systems for Sampling

(focus on *local* dynamics, in *continuous time* (makes life easier)**)**

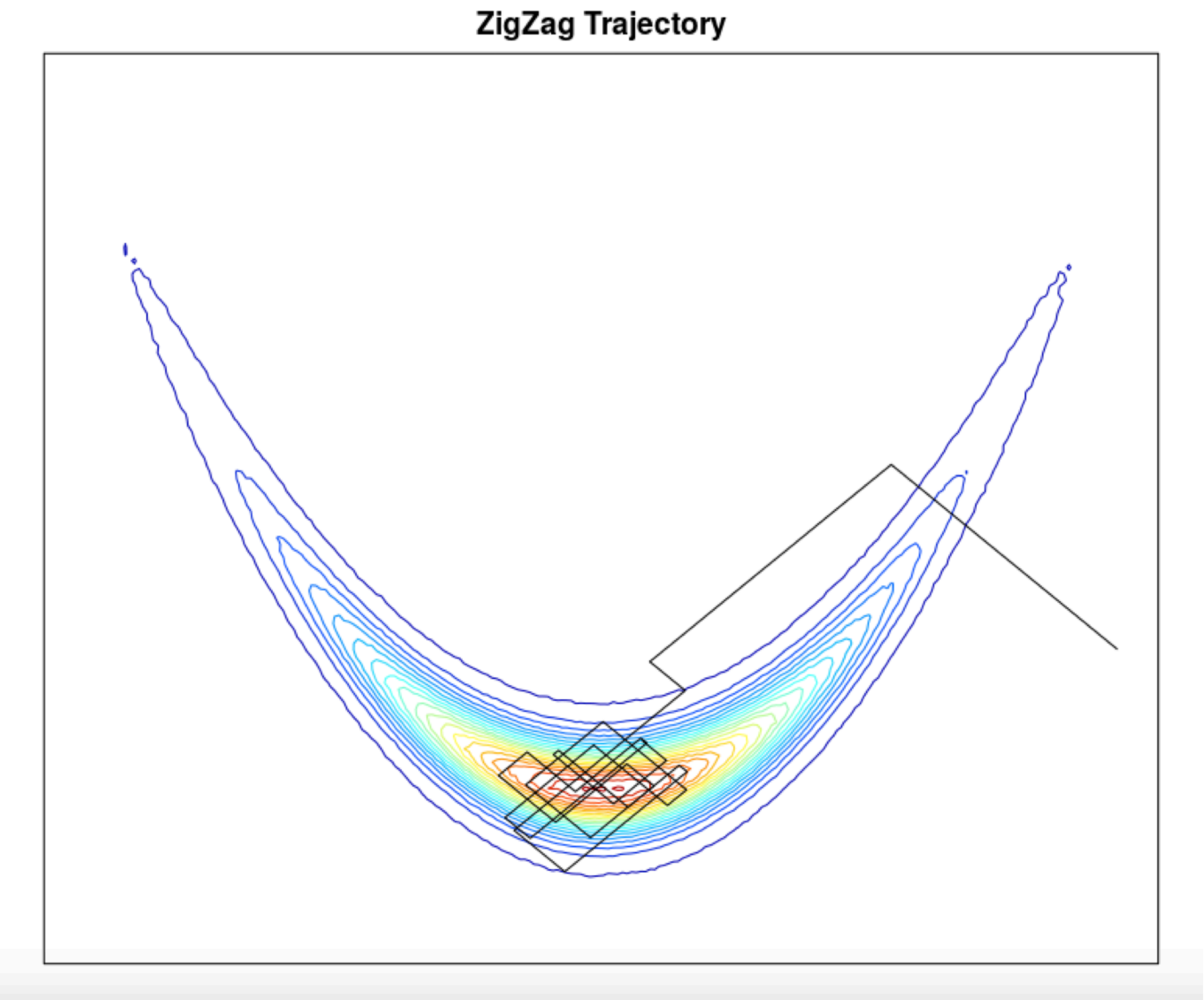
- what can dynamics involve?
    1. following a vector field
    2. diffusing locally according to some metric
    3. 'structured' jumps
- have some fun: hybrid strategies
    - 1 = ?, 1 + 2 = ?, 1 + 3 = ?, 1 + 2 + 3 = ?

Dynamical Systems for Sampling

(focus on *local* dynamics, in *continuous time* (makes life easier)**)**

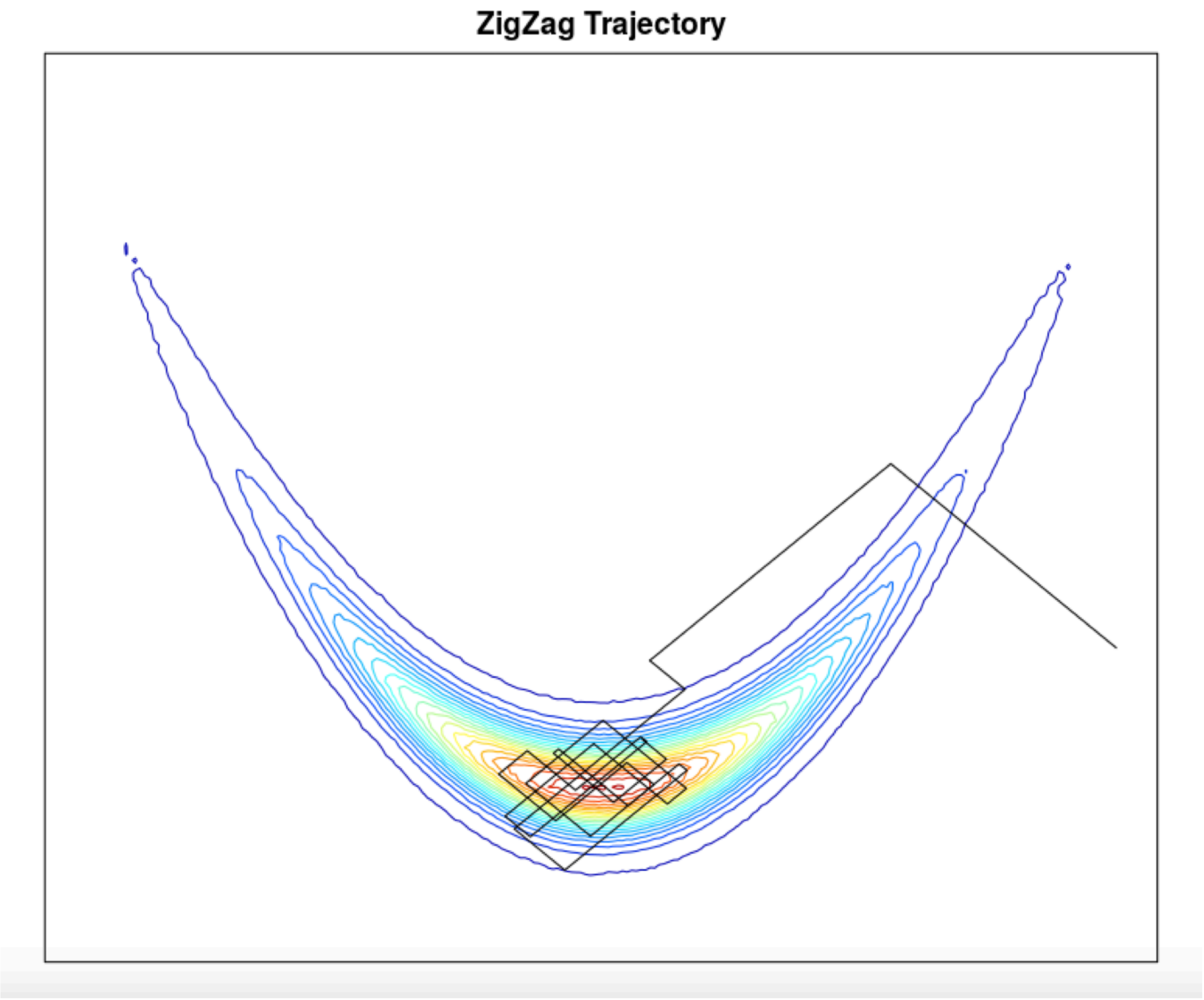
- what can dynamics involve?
    1. following a vector field
    2. diffusing locally according to some metric
    3. 'structured' jumps
- have some fun: hybrid strategies
    - 1 = ?, 1 + 2 = ?, 1 + 3 = ?, 1 + 2 + 3 = ?
        - today: PDMPs = 1 + 3

Primer on PDMPs



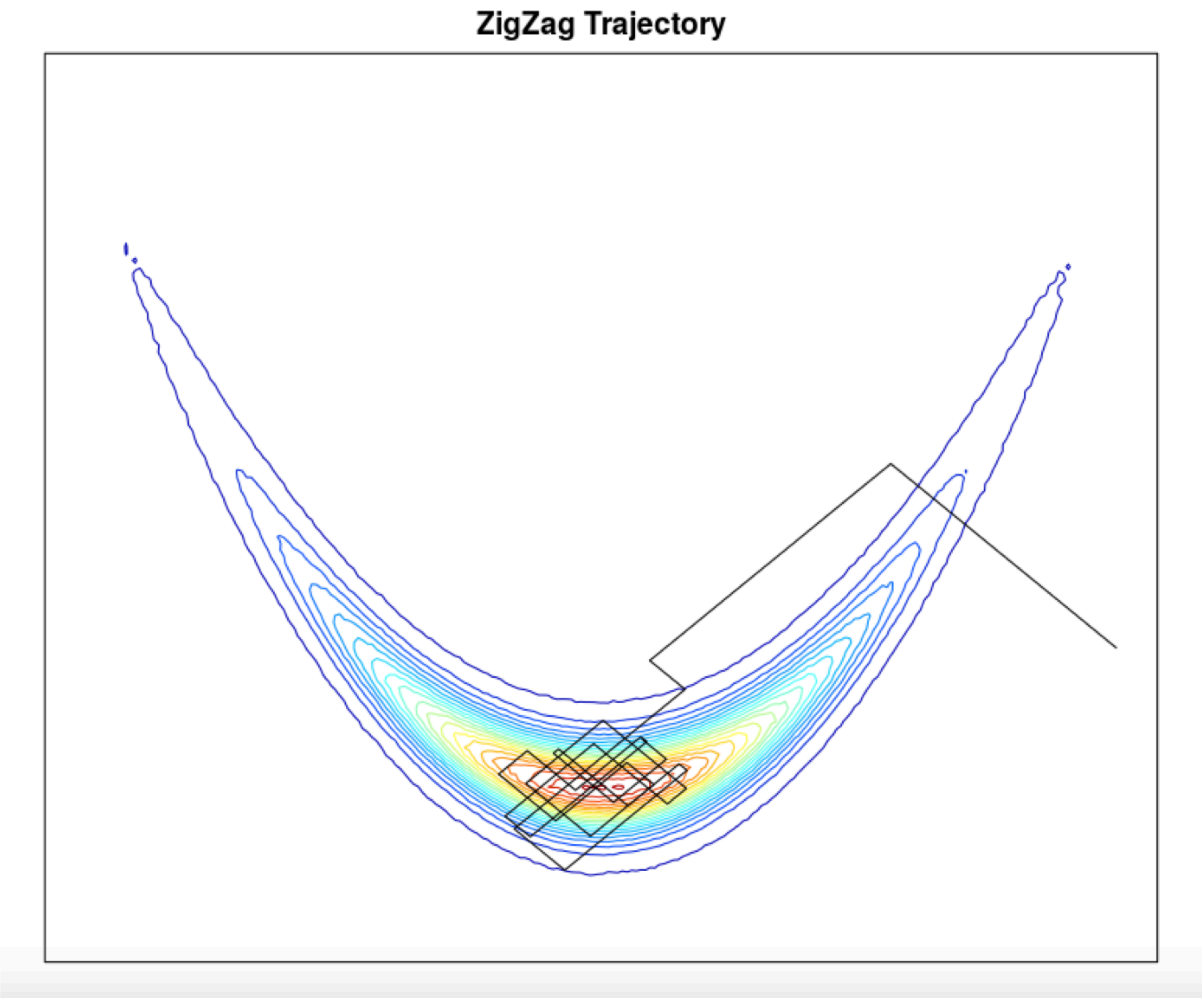
Primer on PDMPs

- ‘Piecewise-Deterministic Markov Processes’ need



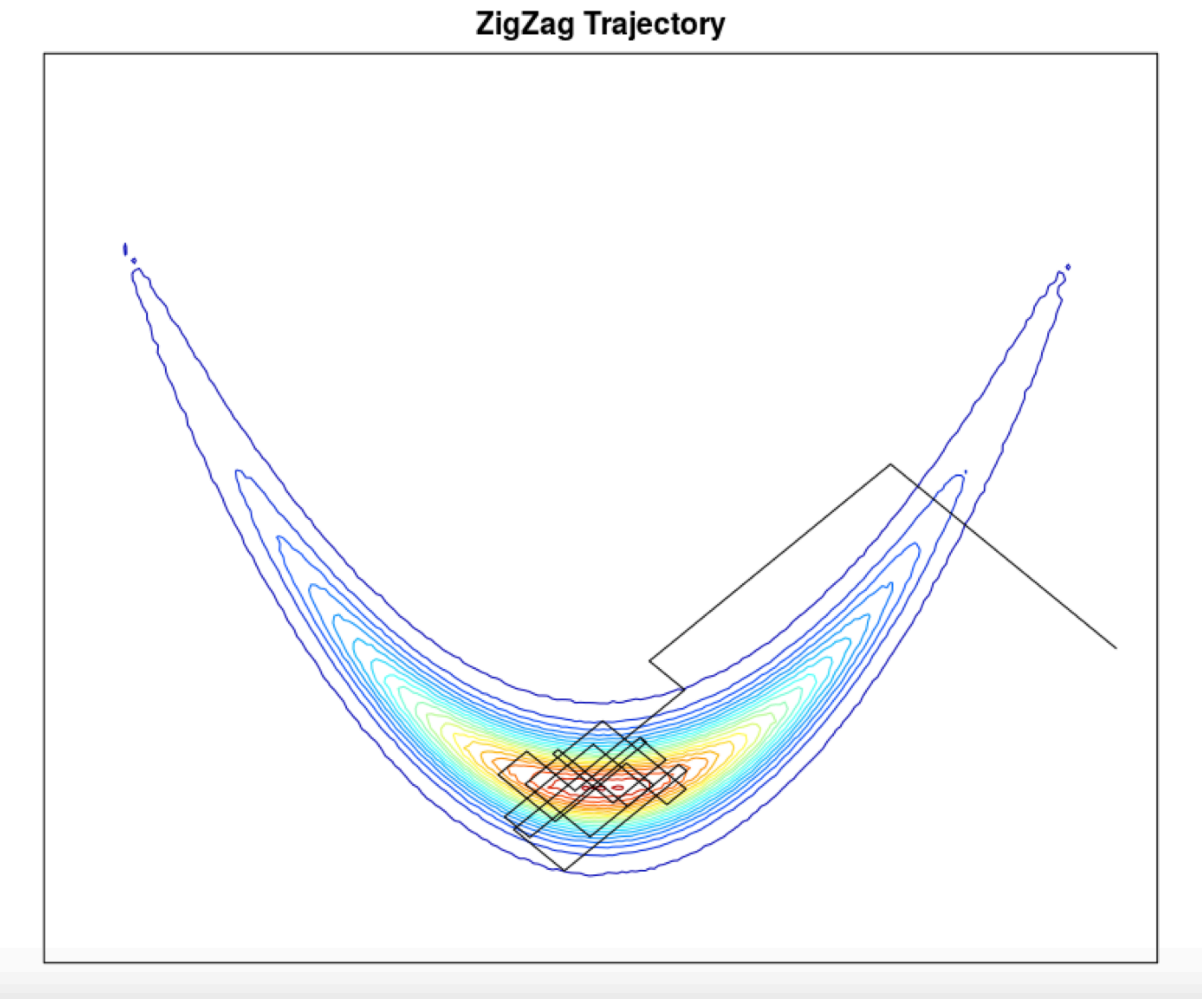
Primer on PDMPs

- ‘Piecewise-Deterministic Markov Processes’ need
  - a vector field



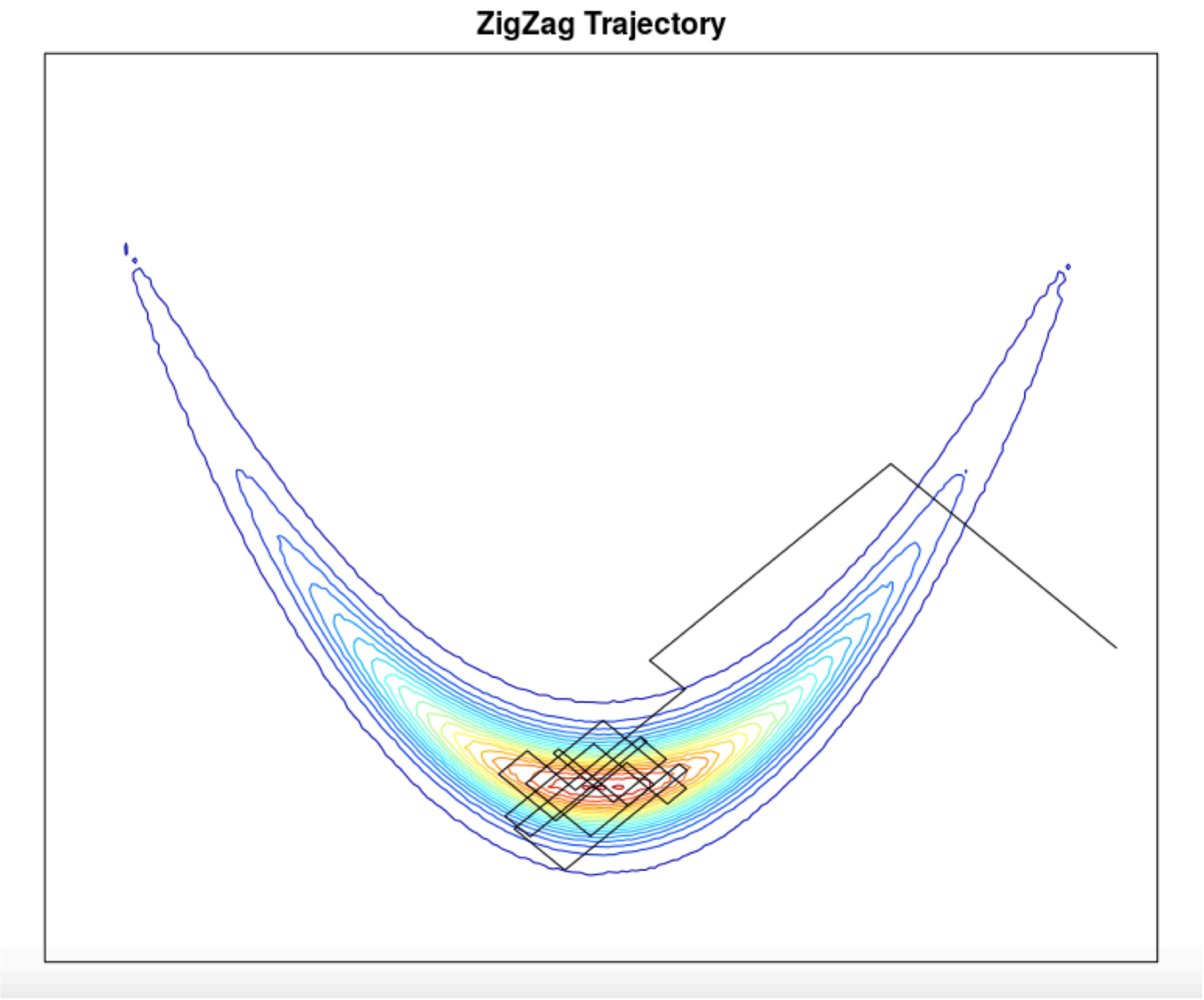
Primer on PDMPs

- ‘Piecewise-Deterministic Markov Processes’ need
  - a vector field
  - an ‘event rate’



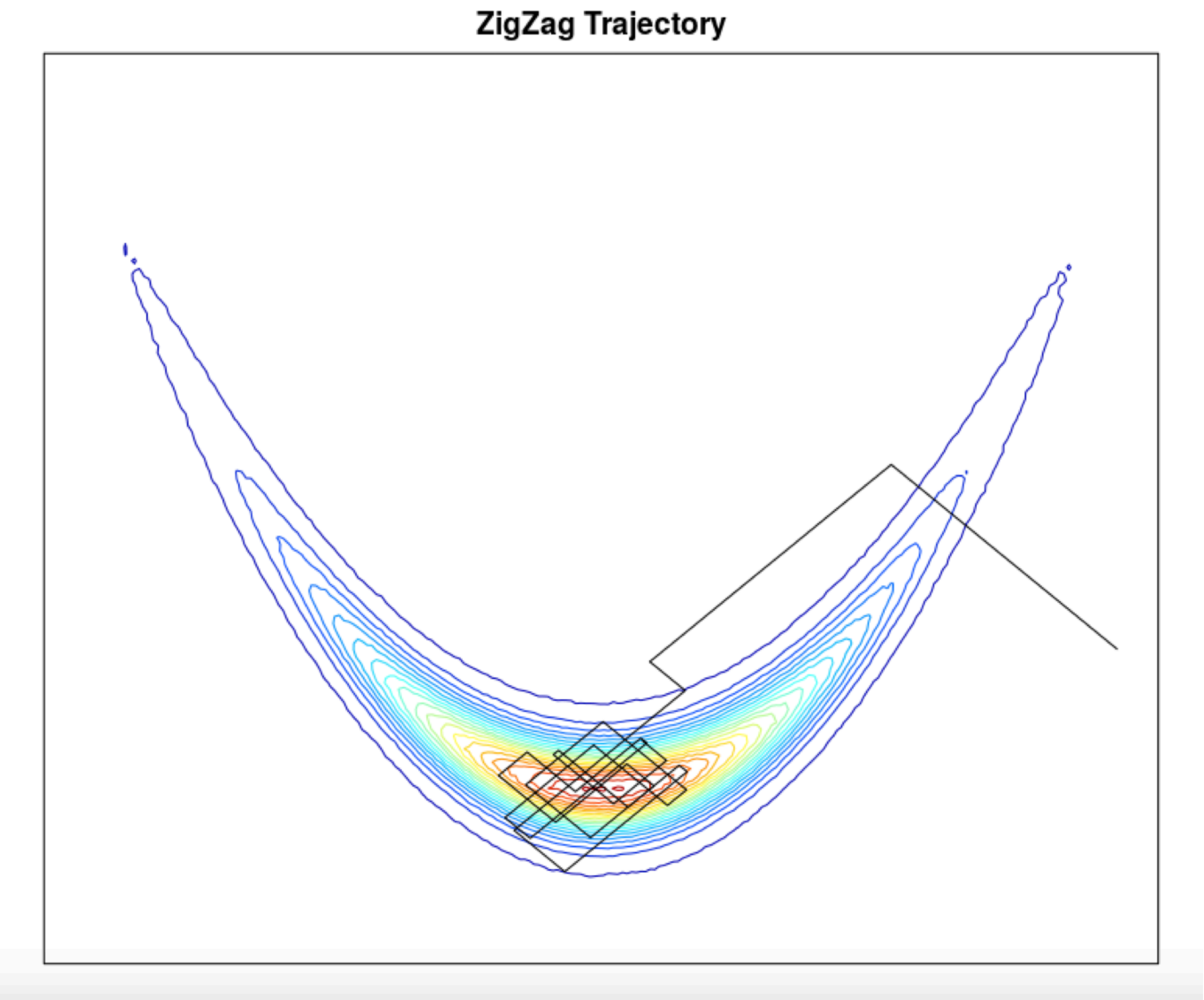
Primer on PDMPs

- 'Piecewise-Deterministic Markov Processes' need
  - a vector field
  - an 'event rate'
  - a jump kernel



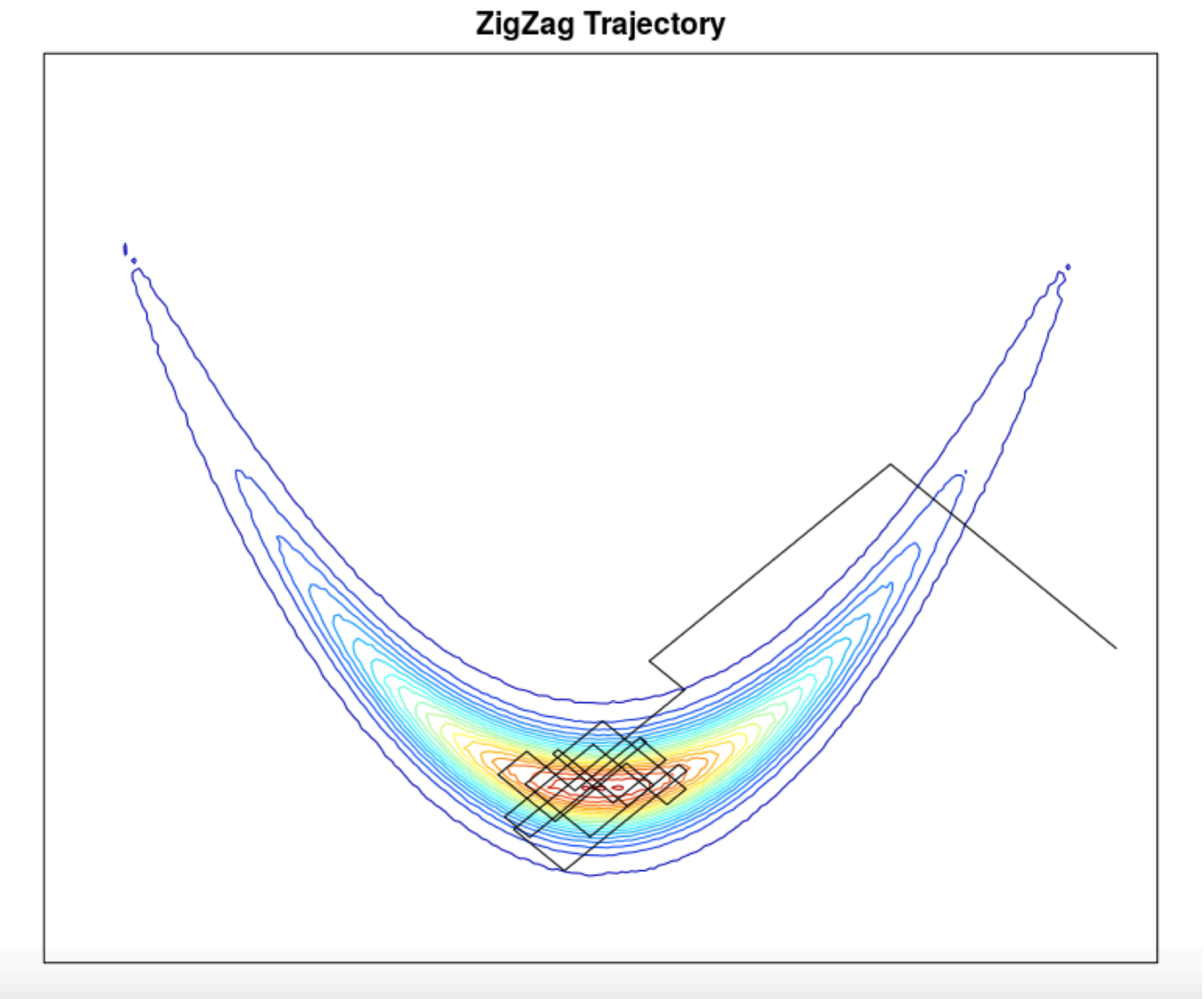
Primer on PDMPs

- ‘Piecewise-Deterministic Markov Processes’ need
  - a vector field
  - an ‘event rate’
  - a jump kernel
- and then ….



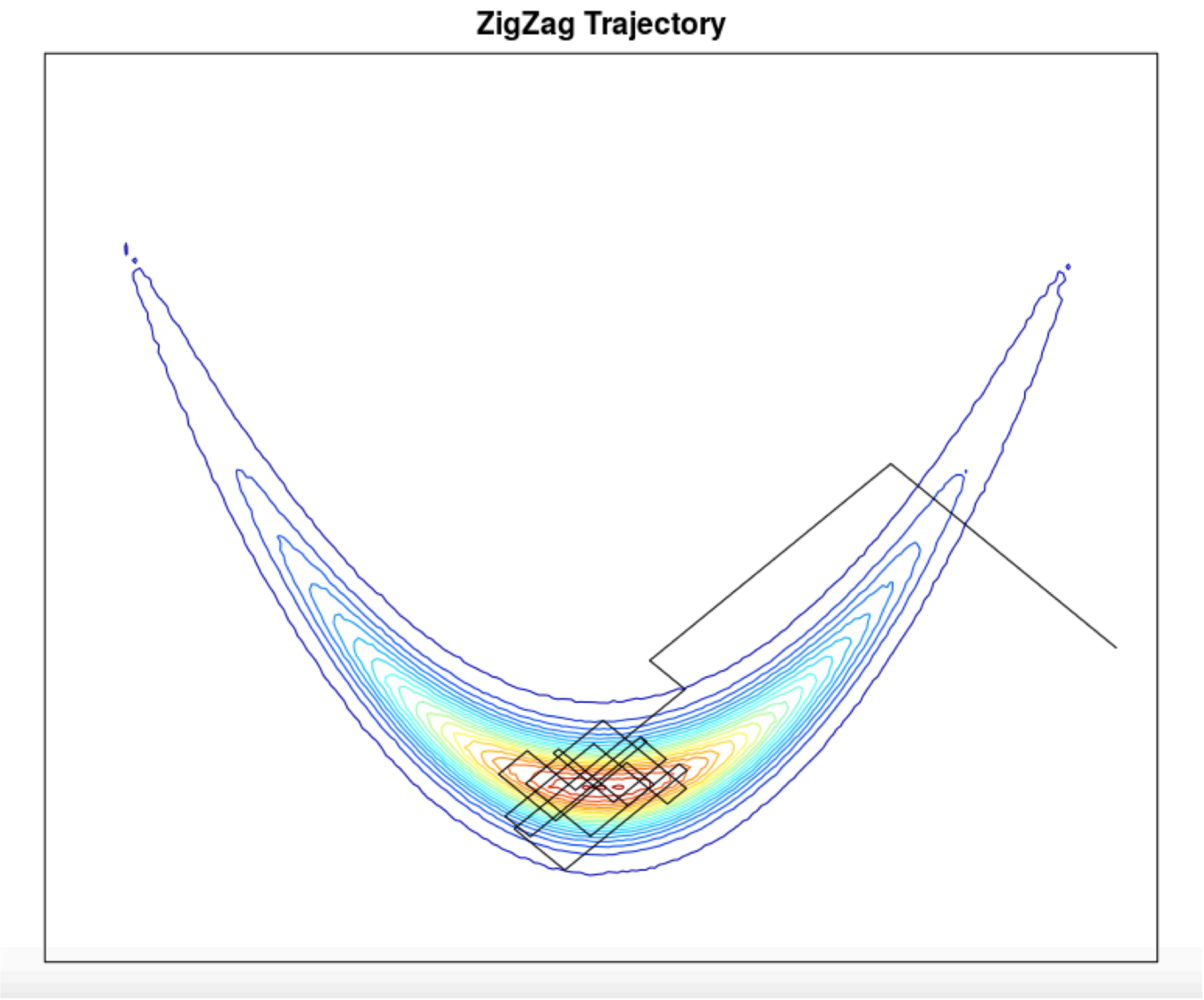
Primer on PDMPs

- ‘Piecewise-Deterministic Markov Processes’ need
  - a vector field
  - an ‘event rate’
  - a jump kernel
- and then ….
  - follow vector field



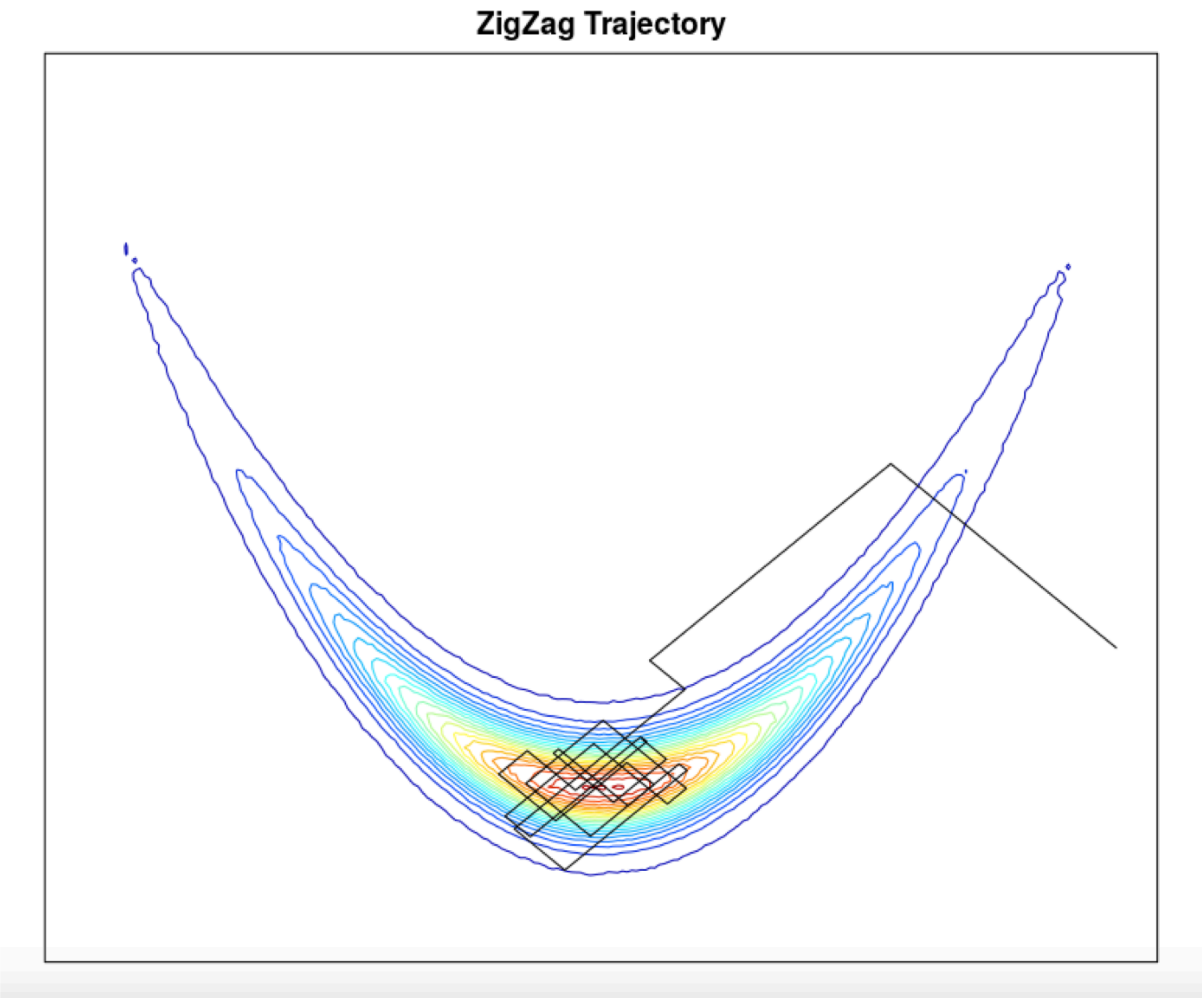
Primer on PDMPs

- ‘Piecewise-Deterministic Markov Processes’ need
  - a vector field
  - an ‘event rate’
  - a jump kernel
- and then ….
  - follow vector field
  - experience event



Primer on PDMPs

- 'Piecewise-Deterministic Markov Processes' need
  - a vector field
  - an 'event rate'
  - a jump kernel
- and then ….
  - follow vector field
  - experience event
  - perform jump



Examples of PDMPs for Monte Carlo

(today: just free transport ($\dot{x} = v, \dot{v} = 0$), others exist)

Examples of PDMPs for Monte Carlo

(today: just free transport ($\dot{x} = v, \dot{v} = 0$), others exist)

1. ‘Bouncy Particle Sampler’

    - events at rate $\lambda(x, v) = \max\left(0, \langle v, \nabla U(x) \rangle\right)$

    - jumps: specular reflection of v along contours of U

Examples of PDMPs for Monte Carlo

(today: just free transport ($\dot{x} = v, \dot{v} = 0$), others exist)

2. ‘Zig-Zag Process’

- events at rate $\lambda_i(x, v) = \max\left(0, \langle v_i, \nabla_{x_i} U(x)\rangle\right)$
- jumps: flip v_i

Examples of PDMPs for Monte Carlo

(today: just free transport ($\dot{x} = v, \dot{v} = 0$), others exist)

3. 'Local' Bouncy Particle Sampler

- assume structured target: $\pi(x) = \exp\left(-\sum_{a} U_a(x_{\partial a})\right)$
- events at rate $\lambda_a(x, v) = \max\left(0, \langle v_{\partial a}, \nabla U_a\left(x_{\partial a}\right)\rangle\right)$
- jumps: specular reflection of $v_{\partial a}$ along contours of U_a

Design of PDMP Methods

Design of PDMP Methods

- general characterisation of invariant PDMPs: too broad!
    - c.f. all invariant jump processes

Design of PDMP Methods

- general characterisation of invariant PDMPs: too broad!
  - c.f. all invariant jump processes
- meaningful restriction: invariant **and** ‘skew-reversible’

Design of PDMP Methods

- general characterisation of invariant PDMPs: too broad!
  - c.f. all invariant jump processes
- meaningful restriction: invariant **and** ‘skew-reversible’
- abstraction: ‘time-enriched’ PDMP
  - process is $(x, \tau) \in \mathbb{R}^d \times \{\pm 1\}$
  - dynamics: $\dot{x} = \tau \cdot b(x)$
  - τ ~ ‘direction of time’; at jumps, flip τ

Design of PDMP Methods

- general characterisation of invariant PDMPs: too broad!
  - c.f. all invariant jump processes
- meaningful restriction: invariant **and** ‘skew-reversible’
- abstraction: ‘time-enriched’ PDMP
  - process is $(x, \tau) \in \mathbb{R}^d \times \{\pm 1\}$
  - dynamics: $\dot{x} = \tau \cdot b(x)$
  - τ ~ ‘direction of time’; at jumps, flip τ
- skew-reversibility: backwards in time, process looks *almost* the same

Basic Invariance Result

Basic Invariance Result

- Define ‘natural event rate’ $r(x) = \langle b(x), \nabla U(x) \rangle - \mathrm{div} b(x)$

Basic Invariance Result

- Define ‘natural event rate’ $r(x) = \langle b(x), \nabla U(x) \rangle - \mathrm{div} b(x)$
- Set $\lambda(x, \tau) = \max \left(0, \tau \cdot r(x)\right) + \gamma(x)$

Basic Invariance Result

- Define 'natural event rate' $r(x) = \langle b(x), \nabla U(x) \rangle - \mathrm{div} b(x)$
- Set $\lambda(x, \tau) = \max\left(0, \tau \cdot r(x)\right) + \gamma(x)$
- Let $Q^{\tau}(x \to \cdot\,)$ be reversible w.r.t. $\pi(x) \cdot \lambda(x, \tau)$

Basic Invariance Result

- Define 'natural event rate' $r(x) = \langle b(x), \nabla U(x) \rangle - \mathrm{div} b(x)$
- Set $\lambda(x, \tau) = \max\left(0, \tau \cdot r(x)\right) + \gamma(x)$
- Let $Q^\tau(x \to \cdot\,)$ be reversible w.r.t. $\pi(x) \cdot \lambda(x, \tau)$
- **Theorem**: necessary and sufficient for π-skew-reversibility

Basic Invariance Result

- Define 'natural event rate' $r(x) = \langle b(x), \nabla U(x) \rangle - \mathrm{div} b(x)$
- Set $\lambda(x, \tau) = \max\left(0, \tau \cdot r(x)\right) + \gamma(x)$
- Let $Q^{\tau}(x \to \cdot\,)$ be reversible w.r.t. $\pi(x) \cdot \lambda(x, \tau)$
- **Theorem**: necessary and sufficient for π-skew-reversibility
    - invariance follows

Structured Enhancement

Structured Enhancement

- recall ZZ, LBPS: event types are tailored to coordinates, target structure

Structured Enhancement

- recall ZZ, LBPS: event types are tailored to coordinates, target structure
- generalisation: *split* time-enriched PDMP
  - dynamics: $\dot{x} = \sum_{\ell} \tau_\ell b_\ell(x),\ \tau_\ell \in \{\pm 1\}$
  - decompose $r(x, \tau) = \sum_j r_j(x, \tau)$
    - each term antisymmetric under a certain 'flip' $\mathscr{F}_j$ of τ variables
    - at 'j-jump', apply j^{th} flip operator

Structured Invariance Result

Structured Invariance Result

- Set $\lambda_j(x, \tau) = \max\left(0, r_j(x, \tau)\right) + \gamma_j(x, \tau)$
    - require γ_j invariant under $\tau \mapsto -\tau$, $\tau \mapsto \mathcal{F}_j\tau$

Structured Invariance Result

- Set $\lambda_j(x,\tau) = \max\left(0, r_j(x,\tau)\right) + \gamma_j(x,\tau)$
  - require γ_j invariant under $\tau \mapsto -\tau$, $\tau \mapsto \mathscr{F}_j \tau$
- Let $Q_j^\tau(x \to \cdot\,)$ be reversible w.r.t. $\pi(x) \cdot \lambda_j(x,\tau)$
  - require $Q_j^\tau = Q_j^{-\mathscr{F}_j \tau}$

Structured Invariance Result

- Set $\lambda_j(x,\tau) = \max\left(0, r_j(x,\tau)\right) + \gamma_j(x,\tau)$
  - require γ_j invariant under $\tau \mapsto -\tau$, $\tau \mapsto \mathscr{F}_j\tau$
- Let $Q_j^{\tau}(x \to \cdot\,)$ be reversible w.r.t. $\pi(x) \cdot \lambda_j(x,\tau)$
  - require $Q_j^{\tau} = Q_j^{-\mathscr{F}_j\tau}$
- **Theorem**: necessary and sufficient for π-skew-reversibility
  - invariance follows

Conclusions

Conclusions

- dynamical systems provide natural principles for sampling algorithms

Conclusions

- dynamical systems provide natural principles for sampling algorithms
  - drift, diffuse, jump!

Conclusions

- dynamical systems provide natural principles for sampling algorithms
  - drift, diffuse, jump!
- can characterise π-invariant piecewise-deterministic Markov processes

Conclusions

- dynamical systems provide natural principles for sampling algorithms
    - drift, diffuse, jump!
- can characterise π-invariant piecewise-deterministic Markov processes
    - structured { targets / dynamics / events } easier to analyse

Conclusions

- dynamical systems provide natural principles for sampling algorithms
  - drift, diffuse, jump!
- can characterise π-invariant piecewise-deterministic Markov processes
  - structured { targets / dynamics / events } easier to analyse
  - simplifies design of new methods

Conclusions

- dynamical systems provide natural principles for sampling algorithms
  - drift, diffuse, jump!
- can characterise π-invariant piecewise-deterministic Markov processes
  - structured { targets / dynamics / events } easier to analyse
  - simplifies design of new methods
- ongoing: simplifying implementation

thanks!

Design of Sampling Dynamics

(ODE, SDE case)

- bare minimum: dynamics should leave π invariant
  - linear condition on generator of process (simple to check, in principle)
- for pure ODE: specify skew-symmetric $Q(x)$, then define
  - $b(x) = Q(x)\,\nabla \log \pi(x) + \mathrm{div} Q(x)$
  - evolve by $\dot{x} = b(x)$
- for reversible SDE: specify PSD $D(x)$, then define
  - $b(x) = D(x)\,\nabla \log \pi(x) + \mathrm{div} D(x)$
  - evolve by $\mathrm{d}x = b(x)\,\mathrm{d}t + \sqrt{2D(x)}\mathrm{d}W$

Design of Sampling Dynamics

(ODE, SDE case)

- general SDE = (pure ODE) + (reversible SDE)
  - so: pick $\left(Q(x), D(x)\right)$, and you're good to go!
- result is ~ old:
  - versions known in stat phys since ~1970s
  - re-popularised by Ma-Chen-Wu-Fox in ML (stochastic gradients)
  - expanded by Barp-Betancourt-Takao-Arnaudon-Girolami (geometry)
- but … PDMPs?