# Sampling from feasible regions of semi-definite programs

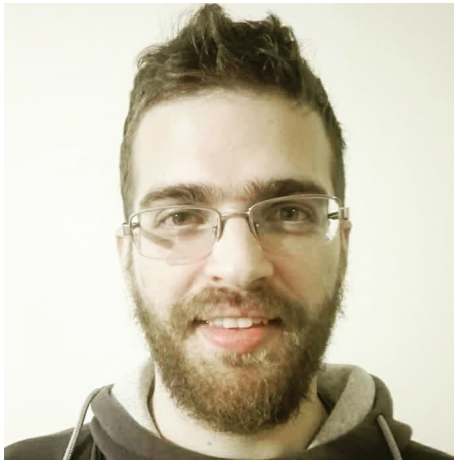
Elias TSIGARIDAS

Inria Paris and IMJ-PRG Sorbonne Université

# Outline

# Outline

# GeomScale org and collaborators



Tolis Chalkis

Quantagonia

Vissarion Fisikopoulos

Oracle

Marios Papachristou

Cornell

But also

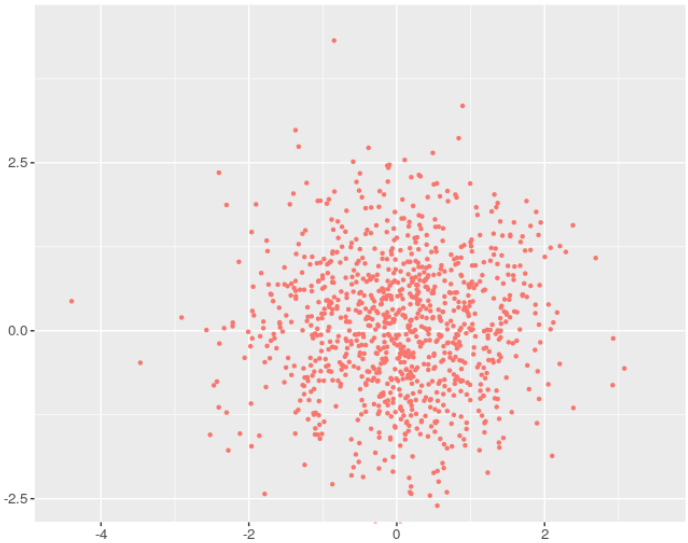
Veni Arakelian, Cyril Bachelard, Ioannis Emiris, Haris Zafeiropoulos, ...

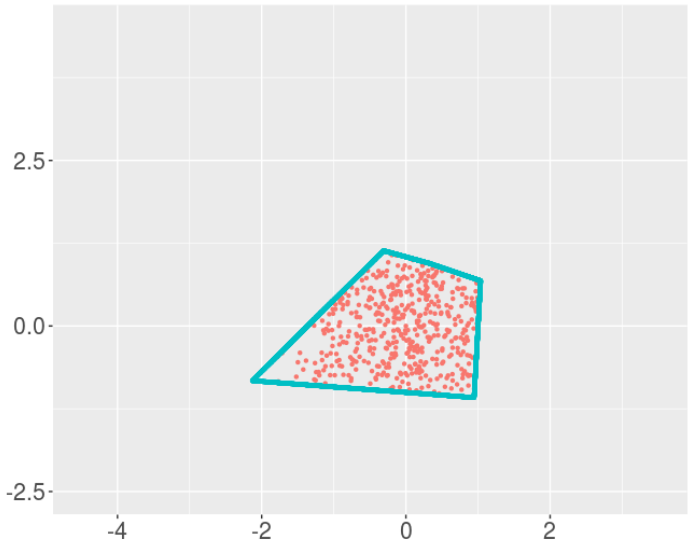
# Outline

# (Truncated) distributions

- Multivariate probability distribution with density function $\pi(x)$
- Truncate $\pi$ to a polytope $P := \{Ax \leq b\}$ we obtain p.d.f. $\pi_P$

$$\pi_P(x) = \frac{f(x)\pi(x)}{\int_P \pi(x)dx}, \quad f(x) = \begin{cases} 1, & \text{if } x \in P \\ 0, & \text{if } x \notin P \end{cases}$$




*The support is the polytope $P$ and $\pi_P$ is the uniform ditribution over $P$.*
*In general the support is a convex body $K \subset \mathbb{R}^n$*

## Problem

Sample (efficiently?) from a (truncated) distribution with density $\pi_K$ ?

## Interesting directions

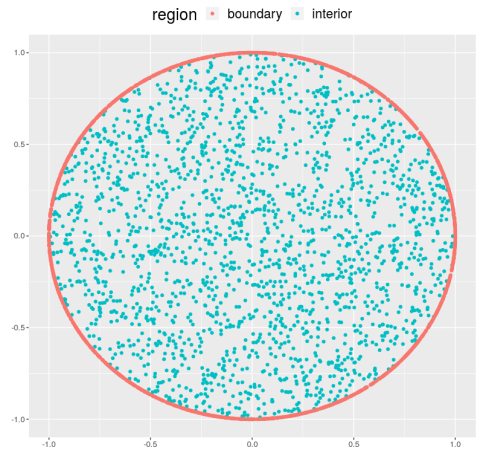
- Algorithms
- Complexity bounds (which computational model? what do we measure?)
- Take advantage of the geometry of support $K \subset \mathbb{R}^n$
  (non-linear, e.g., spectrahedron, basic semi-algebraic set)
- Can we do better when $n$ is small, e.g., $n = 2, 3$ ?
- Applications
  (Volume, integration, Bayesian inference, optimization, ...)

# Outline

# Uniform Sampling from the hypersphere

- To sample uniformly from the boundary of a hypersphere of radius $r$:
  1. Sample $d$ numbers $g_1, \ldots, g_d$ from $\mathcal{N}(0, 1)$.
  2. The point $v = r(g_1, \ldots, g_d)/\sqrt{\sum g_i^2}$ is uniformly distributed on the surface of the $d$-dim hypersphere, of radius $r$ and center the origin.

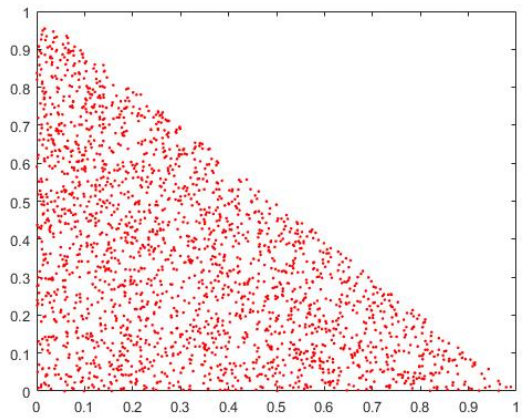
- To sample uniformly from the interior of a hypersphere with radius $r$:
  1. Sample a point $v \sim \mathcal{U}(\partial B_d)$ and $u \sim \mathcal{U}(0, 1)$.
  2. The point $p = ru^{1/d}v$ is uniformly distributed in the interior of the $d$-dim hypersphere, of radius $r$ and center the origin.

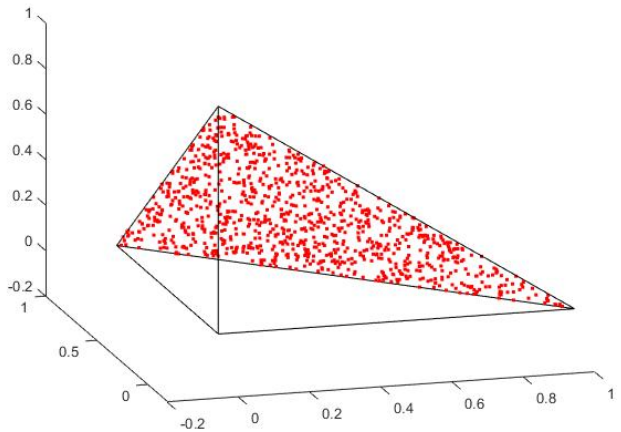



To pick a random direction through point $p \in \mathbb{R}^d$ we sample from the surface of a hypersphere centered at $p$.

# Uniform Sampling from the simplex

1. [Smith, Tromble: 2004]:
   - Generate distinct: $x_0 < x_1 < \cdots < x_{d+1} \in \mathbb{N}^*$.
     Return $y$: $y_i = \dfrac{x_i - x_{i-1}}{M}$, $i = 1, \ldots, d+1$. $M$: largest integer.
   - To guarantee distinct choice we use a variation of Bloom filter.
   - Sampling one point takes $O(d \log d)$.
2. [Rubinstein, Melamed: 1998]:
   - Generate independent unit-exponential random variables, $X_1, \cdots, X_{d+1}$. Return $Y \in \mathbb{R}^{d+1}$: $Y_i = X_i / \sum_{i=1}^{d+1} X_i$.
   - Sampling one point takes $O(d)$.




Generalizations? e.g., Zontopes.

# Outline

# Outline

# Acceptance-rejection sampling

- Let $\pi(x) = f(x)/C,\ x \in \mathbb{R}^d$, where $f(x)$ is an *unnormalized* density and $C \in \mathbb{R}$ a normalizing constant.
- Let $h(x)$ a PDF that can be simulated by some known method and $f(x) \leq kh(x)$, where $k \in \mathbb{R}$ is a constant.

To obtain a sample from $\pi(x)$,

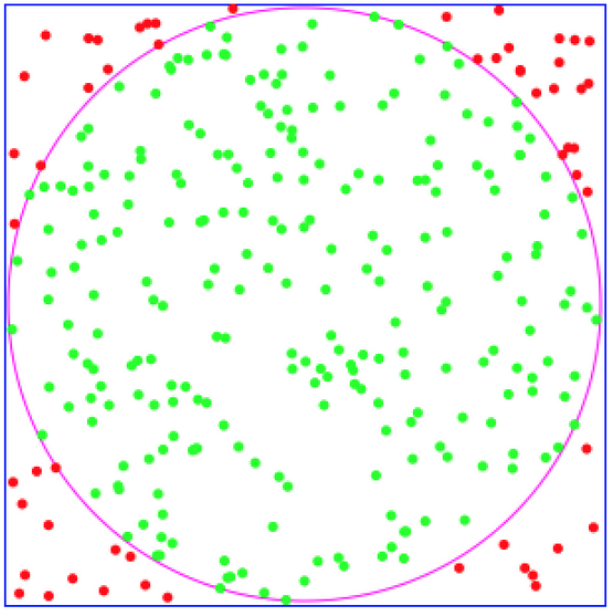
1. Generate a candidate $Z$ from $h(x)$ and a value $u$ from $\mathcal{U}(0,1)$.
2. **If** $u \leq f(Z)/kh(Z)$ **return** Z.
3. **Otherwise** goto 1.

[Flury: 1990]

# Acceptance-rejection sampling

## Drawbacks

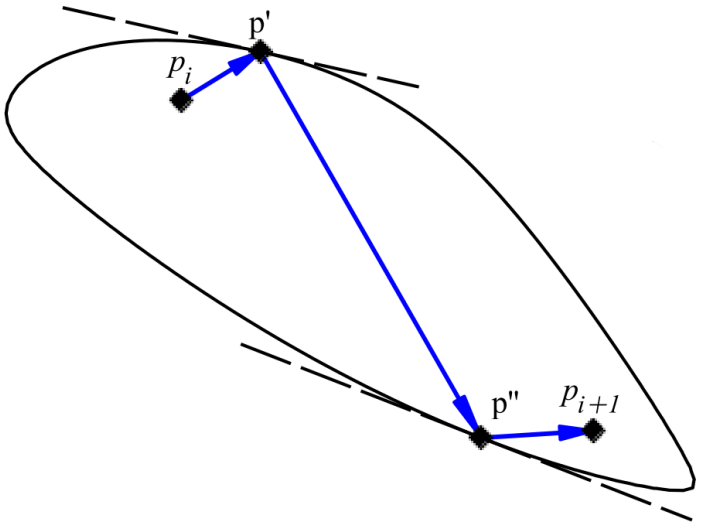
- Sampling/rejections techniques (sample from bounding box) *fail* in high dimensions



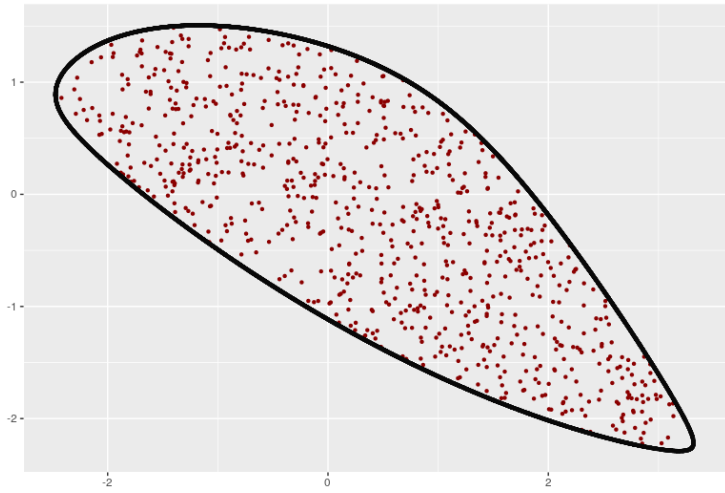
$$\frac{\text{vol}(\textit{unitball})}{\text{vol}(\textit{unitcube})} = O((1/d)^d)$$

# Geometric Random Walks

A Geometric Random Walk starts at some interior point and at each step moves to a "neighboring" point, chosen according to some distribution depending only on the current point.




A Billiard Walk step.

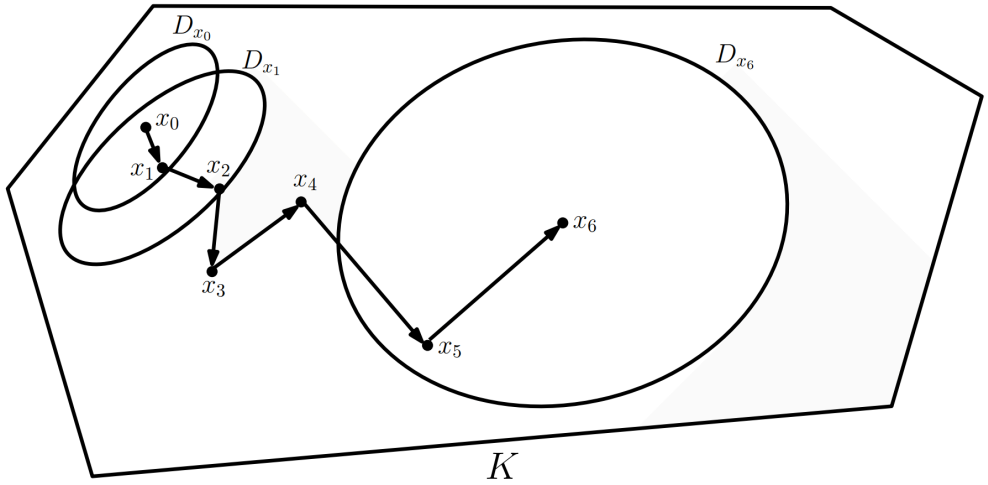



Uniform sampling
(via the Billiard Walk).

# Markov Chain Monte Carlo sampling

A MCMC sampling algorithm applies on a continuous state space $K \subseteq \mathbb{R}^d$

- Starts at a point $x_0 \in K$.
- Being on $x_i$, we move to $x_{i+1}$, according to a transition kernel $p_x(A)$.
- The transition kernel of a MCMC algorithm is the probability to jump from $x$ to a set $A \subseteq K$.
- For example $p_x(K) = 1$.



# Markov Chain Monte Carlo sampling

To sample from a density $\pi(x)$ define a **random walk** on a continuous state space with a transition kernel $p_x(A)$ such that,

1. [Convergence]
$$\int_P p_x(A)\pi(x)dx = \int_A \pi(y)dy$$
Then $\pi(x)$ is called target density.
2. [Uniqueness] $\lim_{n\to\infty} p_x^n(A) = \int_A \pi(y)dy$, where
$$p_x^n(A) = \int_P p_x^{n-1}(y)p_y(A)dy,$$
the transition kernel of the n-th iteration.

[Chib, Greenberg: 1995] *Understanding the Metropolis-Hastings Algorithm*.

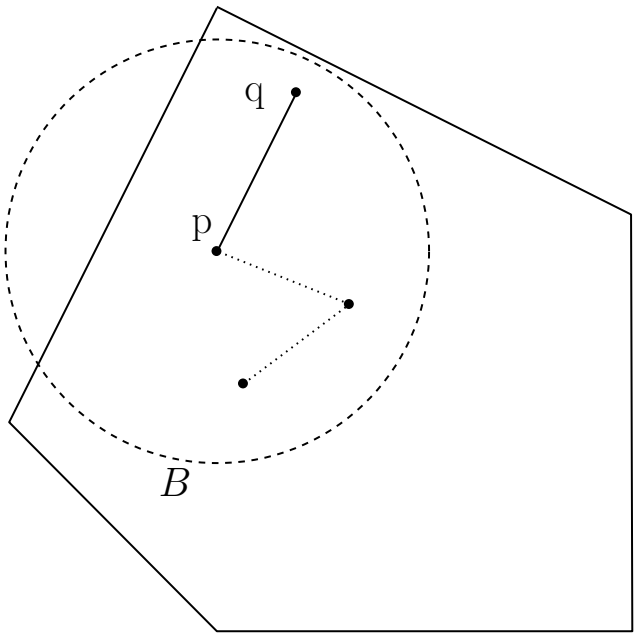
# Useful questions and terminology

- Does the random walk converges asymptotically to the (uniform) distribution? (Correctness)
- How fast does it converge?
  (Equivalently) How many steps do we have to perform until we get a uniform point? (mixing time)
- Does the initial point of the walk affects the efficiency? (warm start)
- What is the cost per step of the random walk?
- Do we assume anything about $K$? (isotropic position, well rounded)

# Ball walk

**Ball Walk**$(K, p, \delta, f)$: convex $K \subset \mathbb{R}^d$, $p \in P$, radius $\delta$, $f : \mathbb{R}^d \to \mathbb{R}_+$

1. Pick a uniform random point $x$ in $B(p, \delta)$.
2. **return** $x$ with probability $\min\left\{1, \frac{f(x)}{f(p)}\right\}$;
   **return** $p$ with the remaining probability.

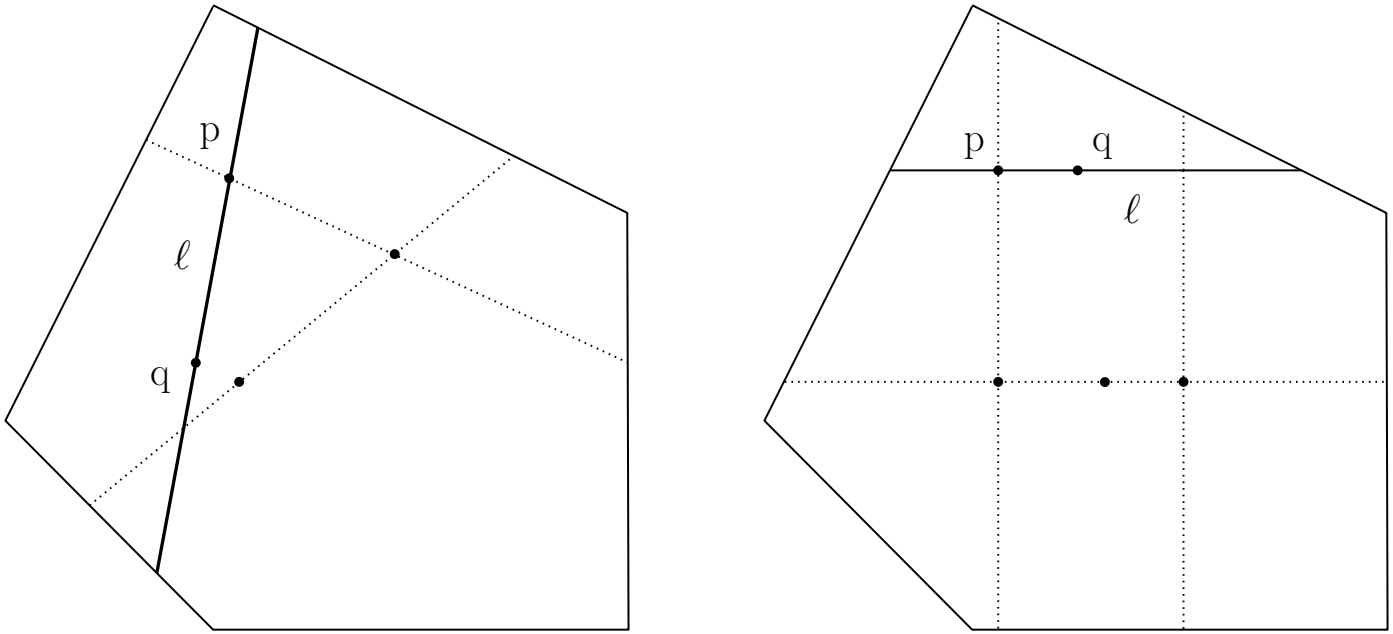



*If the density is not restricted in $K$, then it is the* **Metropolis-Hastings** *algorithm.*

# Hit-and-Run

**Hit and Run**$(K, p, f)$: convex $K \subset \mathbb{R}^d$, point $p \in P$, $f : \mathbb{R}^d \to \mathbb{R}_+$

1. Pick uniformly a line $\ell$ through $p$.
2. **return** a random point on the chord $\ell \cap K$ chosen from the distribution $\pi_{\ell,f}$ restricted in $K \cap \ell$.

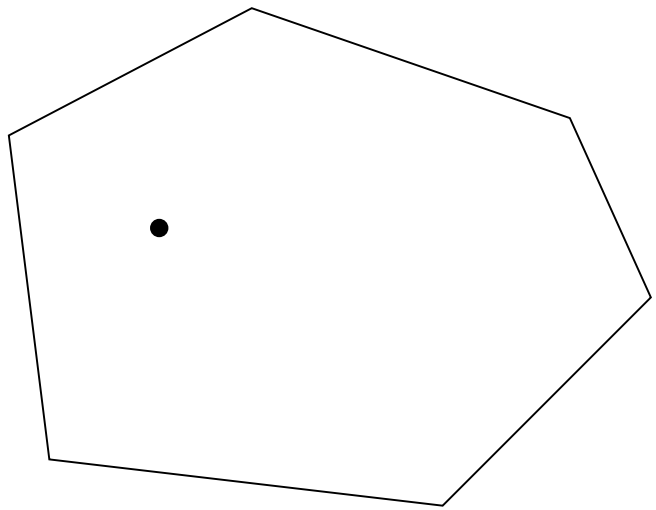



- **Q**: How do we compute $\ell \cap K$? Can we do it *exactly*?

# Billiard walk - Uniform case

**BW**$(K, p_i, \tau, R)$ [Polyak'14]

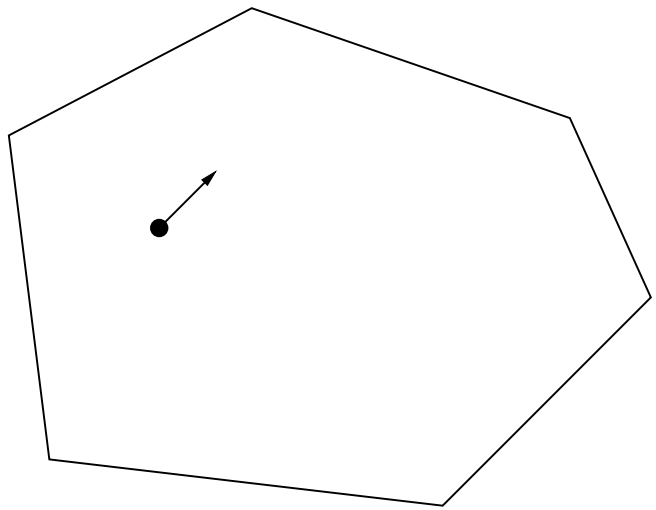
1. Generate the length of the trajectory $L = -\tau \ln \eta$, $\eta \sim U(0,1)$.
2. Pick a uniform direction $v$ to define the trajectory. then the direction becomes $v \leftarrow v - 2\langle v, s \rangle$.
3. If the trajectory meets a boundary with internal normal $s$, $||s|| = 1$,
4. **return** the end of the trajectory as $p_{i+1}$.
   If the number of reflections exceeds $R$, then **return** $p_{i+1} = p_i$.



# Billiard walk - Uniform case

**BW**$(K, p_i, \tau, R)$ [Polyak'14]

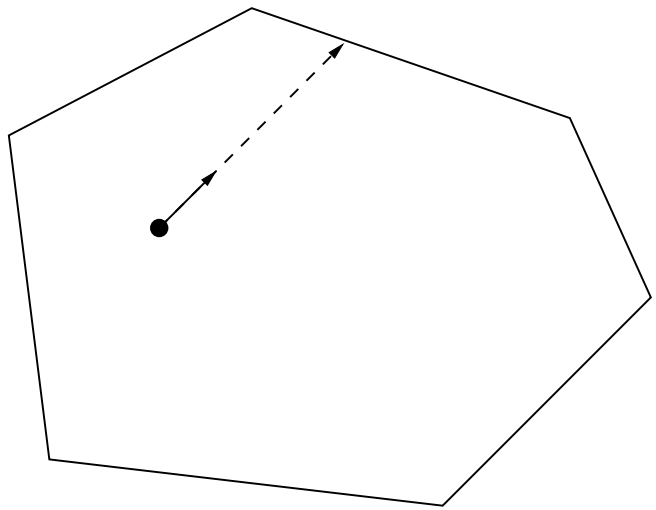
1. Generate the length of the trajectory $L = -\tau \ln \eta$, $\eta \sim U(0,1)$.
2. Pick a uniform direction $v$ to define the trajectory. then the direction becomes $v \leftarrow v - 2\langle v, s \rangle$.
3. If the trajectory meets a boundary with internal normal $s$, $||s|| = 1$,
4. **return** the end of the trajectory as $p_{i+1}$.
   If the number of reflections exceeds $R$, then **return** $p_{i+1} = p_i$.



# Billiard walk - Uniform case

**BW**$(K, p_i, \tau, R)$ [Polyak'14]

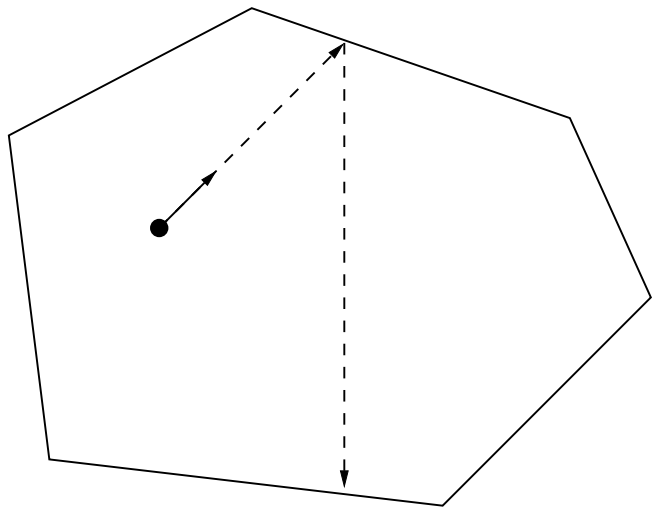
1. Generate the length of the trajectory $L = -\tau \ln \eta$, $\eta \sim U(0,1)$.
2. Pick a uniform direction $v$ to define the trajectory. then the direction becomes $v \leftarrow v - 2\langle v, s \rangle$.
3. If the trajectory meets a boundary with internal normal $s$, $||s|| = 1$,
4. **return** the end of the trajectory as $p_{i+1}$.
   If the number of reflections exceeds $R$, then **return** $p_{i+1} = p_i$.



# Billiard walk - Uniform case

**BW**$(K, p_i, \tau, R)$ [Polyak'14]

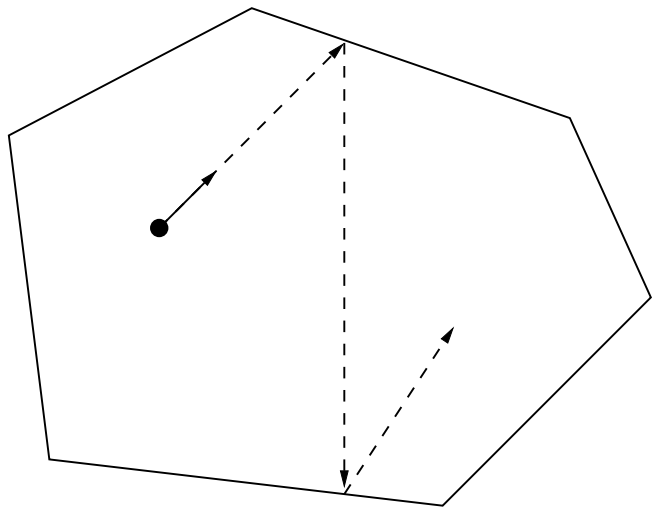
1. Generate the length of the trajectory $L = -\tau \ln \eta$, $\eta \sim U(0,1)$.
2. Pick a uniform direction $v$ to define the trajectory. then the direction becomes $v \leftarrow v - 2\langle v, s \rangle$.
3. If the trajectory meets a boundary with internal normal $s$, $||s|| = 1$,
4. **return** the end of the trajectory as $p_{i+1}$.
   If the number of reflections exceeds $R$, then **return** $p_{i+1} = p_i$.

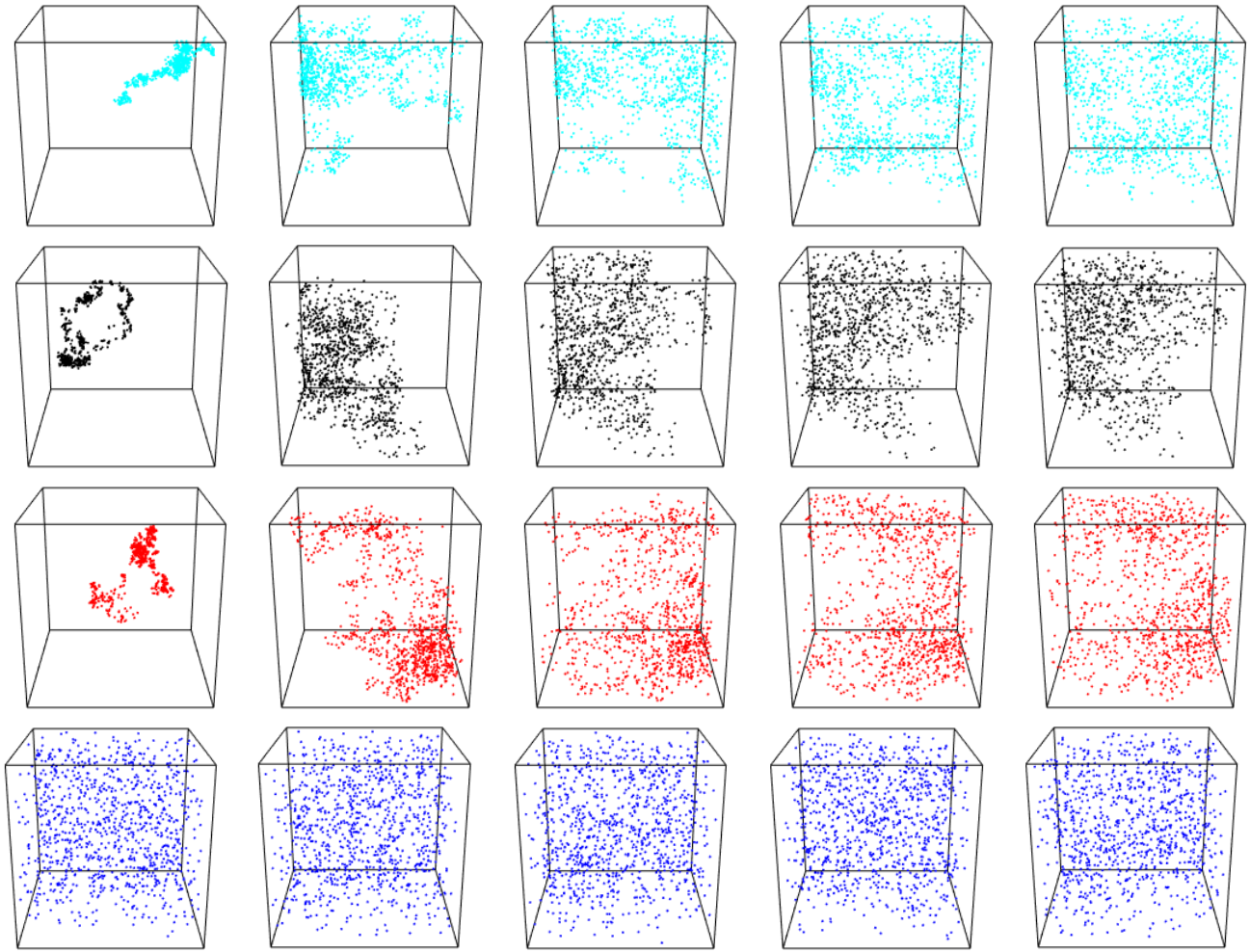


# Billiard walk - Uniform case

**BW**$(K, p_i, \tau, R)$ [Polyak'14]

1. Generate the length of the trajectory $L = -\tau \ln \eta$, $\eta \sim U(0,1)$.
2. Pick a uniform direction $v$ to define the trajectory. then the direction becomes $v \leftarrow v - 2\langle v, s \rangle$.
3. If the trajectory meets a boundary with internal normal $s$, $||s|| = 1$,
4. **return** the end of the trajectory as $p_{i+1}$.
   If the number of reflections exceeds $R$, then **return** $p_{i+1} = p_i$.





- Uniform sampling from the hypercube $[-1, 1]^{200}$ and projection to $\mathbb{R}^3$.
- Rows: Ball Walk, Coordinate Directions Hit and Run, Random Directions Hit and Run, Billiard Walk.
- Columns: walk length, $\{1, 50, 100, 150, 200\}$

# (Some of the) Limitations of BW and HnR

- Their mixing time is $\widetilde{O}(d^3)$ for log-concave distributions.
- Their performance is crucially affected by the starting point.

- Typically, we need a warm start.
  A distribution $S$ is $M$-warm w.r.t. to the distribution $Q$, if

$$M = \sup_{A \in P} \frac{S(A)}{Q(A)}$$

- Better efficiency if the distribution is (approximately) isotropic.
  A distribution $Q$ is is isotropic if

$$\mathbb{E}_Q[X] = 0, \text{ and } \mathbb{E}_Q[XX^T] = I_d$$

# Outline

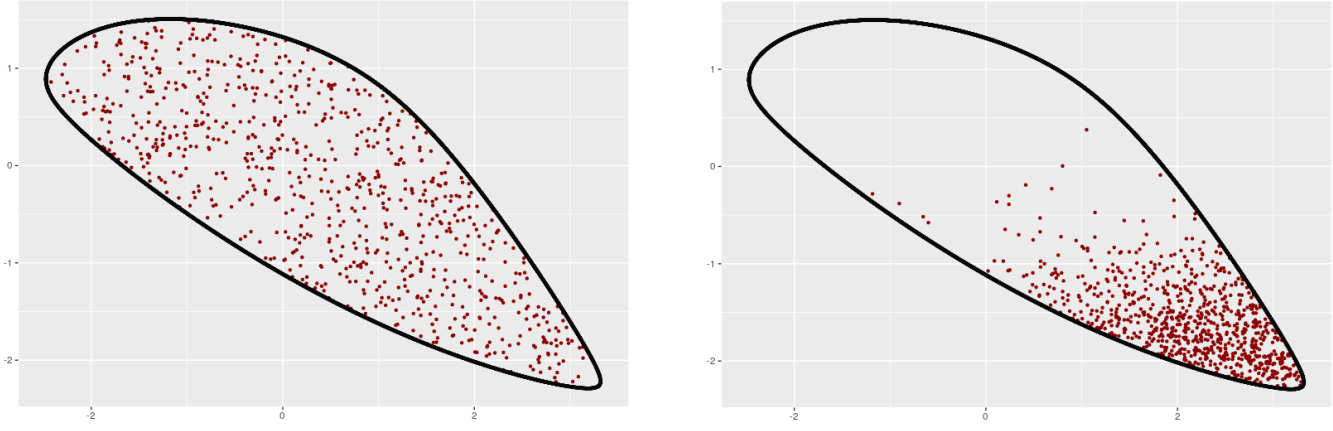
# Outline

# Truncated log-concave sampling

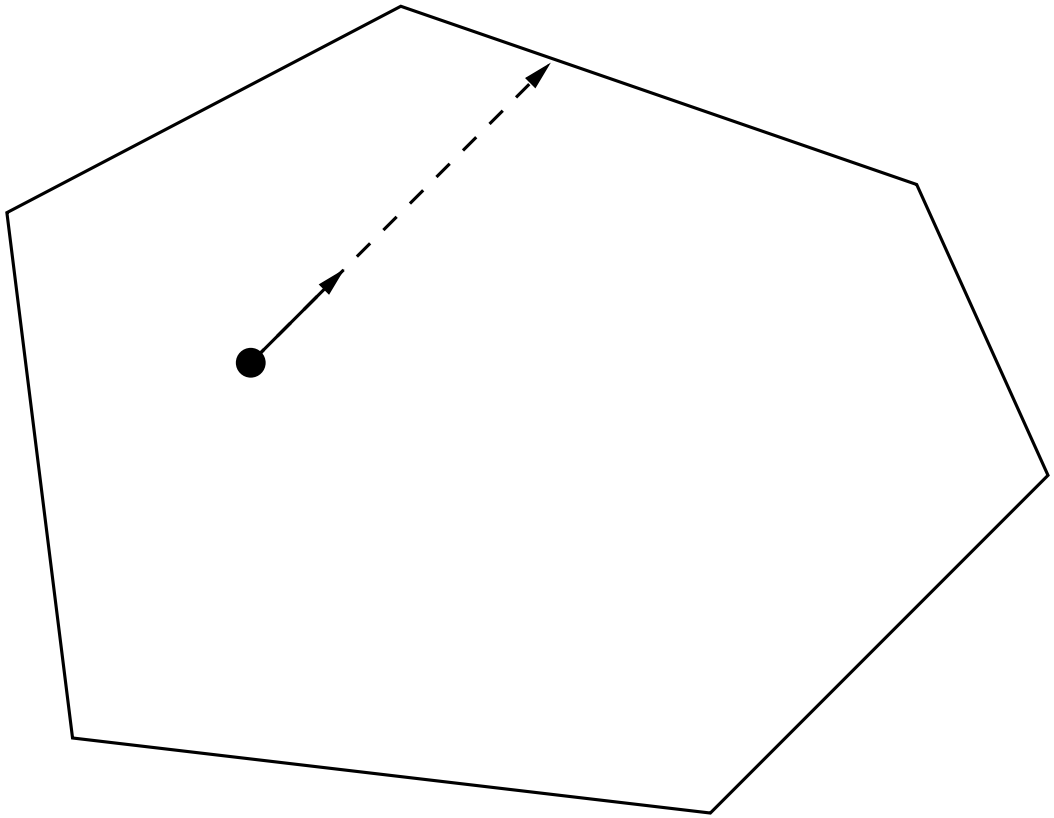
## Definition

Let $\pi(\boldsymbol{x}) \propto e^{-f(\boldsymbol{x})}$, where $f : \mathbb{R}^d \to \mathbb{R}$ is a convex function.
$\pi(\boldsymbol{x})$ is called *log-concave (LC) probability density*.

- Let $\pi(\boldsymbol{x})$ be restricted to **convex body** $K \subset \mathbb{R}^d$.
- Important examples: Uniform, Gaussian, Boltzmann.



# Another view of Billard Walk and Hit-and-Run



# Hamiltonian Monte Carlo

- Being at $\boldsymbol{p} \in K$, HMC introduces an auxiliary random variable $\boldsymbol{v} \in \mathbb{R}^d$ and generates samples from the joint density

$$\pi(\boldsymbol{p}, \boldsymbol{v}) = \pi(\boldsymbol{v}|\boldsymbol{p})\pi(\boldsymbol{p}),$$

- Marginalize out $\boldsymbol{v}$, then recover the target dist. $\pi(\boldsymbol{p})$.
- Consider $\boldsymbol{v} \sim \mathcal{N}(0, I_d)$;
  PDF $\pi(\boldsymbol{p}, \boldsymbol{v}) = e^{-H(\boldsymbol{p},\boldsymbol{v})}$ defines a Hamiltonian,

$$H(\boldsymbol{p}, \boldsymbol{v}) = -\log \pi(\boldsymbol{p}, \boldsymbol{v}) = -\log \pi(\boldsymbol{p}) + \frac{1}{2}|\boldsymbol{v}|^2,$$

# Hamiltonian Monte Carlo

- HMC simulates a particle moving in a conservative field determined by $-\log \pi(\boldsymbol{p})$ and $-\nabla \log \pi(\boldsymbol{p})$.

- HMC, starting from a position $\boldsymbol{p}$, generates a new state:
  1. Draw a value for the momentum, $\boldsymbol{v} \sim \mathcal{N}(0, I_d)$
  2. $(\boldsymbol{p}, \boldsymbol{v})$ is given by the Hamilton's system of ODE:

$$\begin{aligned} \frac{d\boldsymbol{p}}{dt} &= \frac{\partial H(\boldsymbol{p}, \boldsymbol{v})}{\partial \boldsymbol{v}} \\ \frac{d\boldsymbol{v}}{dt} &= -\frac{\partial H(\boldsymbol{p}, \boldsymbol{v})}{\partial \boldsymbol{p}} \end{aligned} \quad \Rightarrow \quad \begin{cases} \frac{d\boldsymbol{p}(t)}{dt} = \boldsymbol{v}(t) \\ \\ \frac{d\boldsymbol{v}(t)}{dt} = -\nabla \log \pi(\boldsymbol{p}) \end{cases} \tag{1}$$

- Solve the ODE using
  – Euler methods (e.g., Leapfrog) [Neal: 2012] or,
  – Collocation method [Vempala,Lee,Song : 2018].

# Solving the ODE

- Usually we use the **leapfrog method**.
- It is a variant of Euler's method:

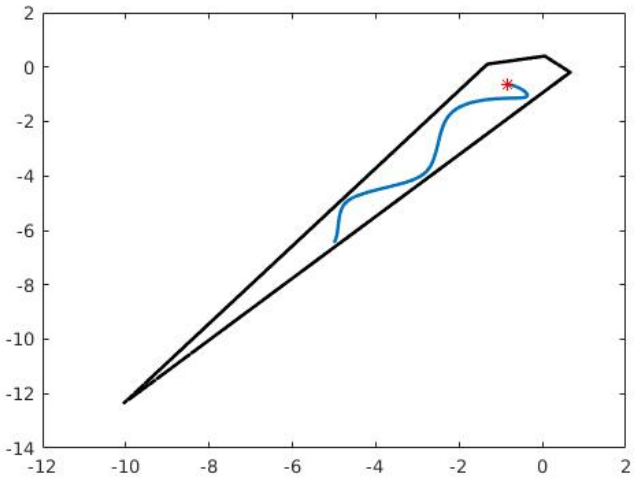
$$\begin{aligned}\mathbf{v}(t+\epsilon/2) &= \mathbf{v}(t) + (\epsilon/2)\nabla \log(\pi(\mathbf{p}(t))) \\ \mathbf{p}(t+\epsilon) &= \mathbf{p}(t) + \epsilon \mathbf{v}(t+\epsilon/2) \\ \mathbf{v}(t+\epsilon) &= \mathbf{v}(t+\epsilon/2) + (\epsilon/2)\nabla \log(\pi(\mathbf{p}(t+\epsilon)))\end{aligned} \tag{2}$$

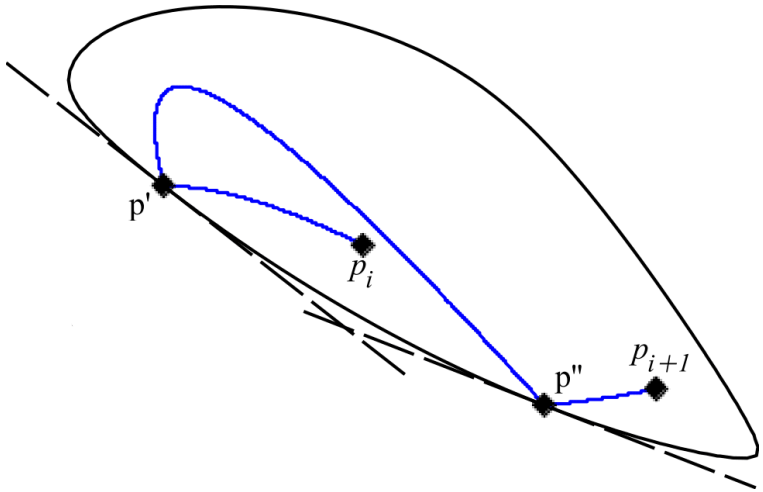
**Question**

What is the bit complexity of leapfrog in our setting?

# Hamiltonian Monte Carlo

- If the density is restricted in $K$, then the trajectories of HMC are in $K$.



# Random walks for truncated log-concave sampling

| Year & Authors | Random walk | Mixing time* | Distribution |
|---|---|---|---|
| [Smith: 1986] | Hit-and-Run | $\widetilde{O}(d^3)$ | any LC |
| [Berbee, Smith: 1987] | Coordinate Hit-and-Run | $\widetilde{O}(d^{10})$ | any LC |
| [Lovasz,Simonovits'90] | Ball walk | $\widetilde{O}(d^3)$ | any LC |
| [Kannan,Narayanan'12] | Dikin walk | $\widetilde{O}(d^2)$ | uniform (H-polytope) |
| [Polyak,Dabbene'14] | **Billiard walk** | ?? | uniform |
| [Afshar,Domke'15] | **Reflective HMC** | ?? | any LC (polytopes) |
| [Lee,Vempala'16] | Geodesic walk | $O(md^{3/4})$ | uniform (H-polytope) |
| [Lee,Vempala'17] | Remannian HMC | $\widetilde{O}(md^{2/3})$ | uniform (H-polytopes) |
| [Chen,Dwivedi,Wainwright,Yu'19] | John walk | $\widetilde{O}(d^{5/2})$ | uniform (H-polytope) |
| [Chen,Dwivedi,Wainwright,Yu'19] | Vaidya walk | $O(m^{1/2}d^{3/2})$ | uniform (H-polytope) |

- Cost per sample: *cost per step* $\times$ *mixing time (#steps)*.
- The *cost per step* depends on the convex body.
- Hit-and-Run (HR): widely used & well studied.
- Coordinate Hit-and-Run (CDHR): seems more efficient than HR in practice.
- Existing software uses either CDHR or HR (H-polytopes).

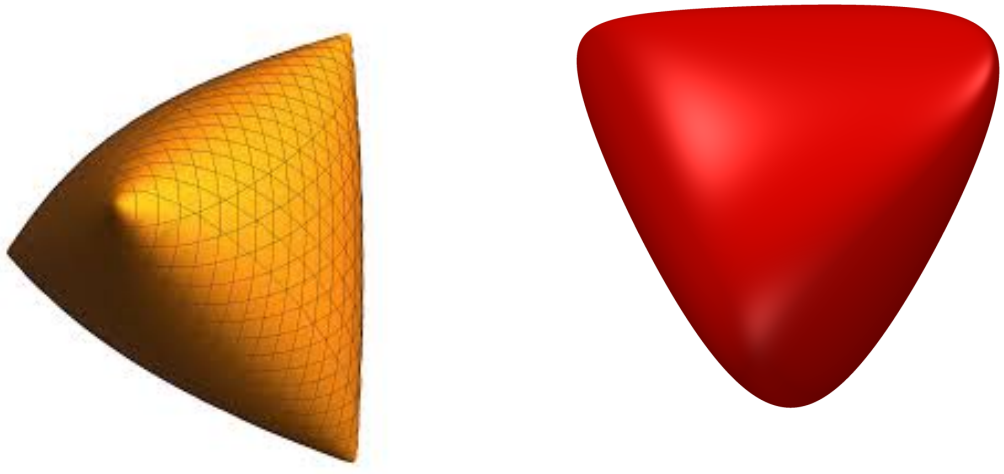
# Outline

## Definition

A **spectrahedron** $S \subset \mathbb{R}^d$ is the feasible set of a **linear matrix inequality**. If $\boldsymbol{A}_i$ are symmetric matrices in $\mathbb{R}^{m \times m}$ and

$$F(\boldsymbol{x}) = \boldsymbol{A}_0 + x_1 \boldsymbol{A}_1 + \cdots + x_d \boldsymbol{A}_d,$$

then $S = \{\boldsymbol{x} \in \mathbb{R}^d \mid F(\boldsymbol{x}) \succeq 0\}$.

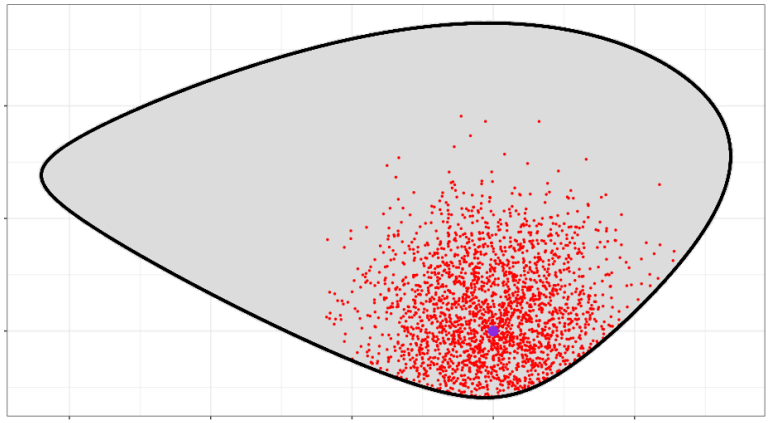


$S$ is the feasible set of a Semidefinite Program (SDP)

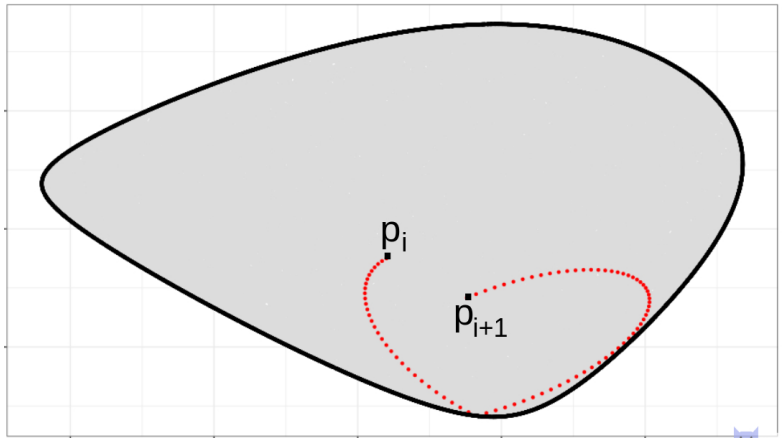
# Reflective Hamiltonian Monte Carlo (ReHMC)

- When the density is restricted in a convex body $K$ then HMC trajectory stays inside $K$ by using boundary reflections.

Case of Leapfrog method



$\pi(x)$




Discrete Hamiltonian trajectory

We pre-select the number of Leapfrog steps

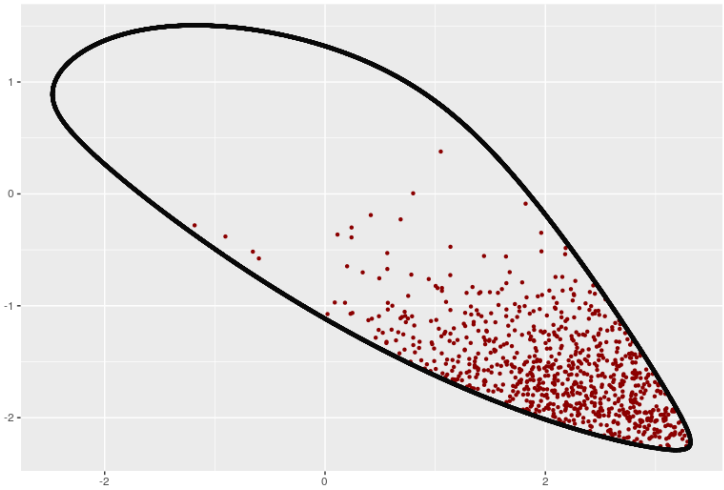
**Theorem** ([Chalkis, Fisikopoulos, Papachristou, T : 2021])

*It converges to the target distribution when $K$ is a spectrahedron.*

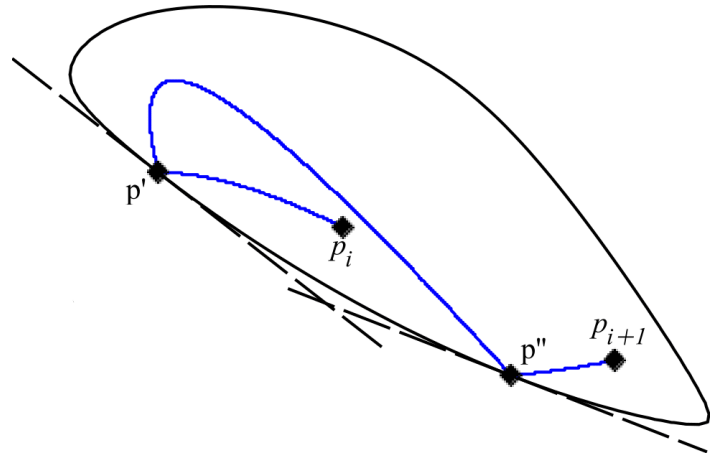
# Reflective Hamiltonian Monte Carlo (ReHMC)

- When the density is restricted in a convex body $K$ then HMC trajectory stays inside $K$ by using boundary reflections.

Case of collocation method



$\pi(x)$




Polynomial Hamiltonian trajectory

We randomly select the integration time in each steps

**Theorem** ([Chalkis, Fisikopoulos, Papachristou, T : 2021])

*ReHMC converges to the target distribution when $K$ is a spectrahedron.*

# Correctness of ReHMC

**Theorem** ([Chalkis, Fisikopoulos, Papachristou, T : 2021])

*For a smoothly differentiable negative log-density $f$, where $\pi \propto \exp(-f(x))$, the discretized reflective Hamiltonian Dynamics are volume-preserving and time-reversible.*

**Theorem** ([Chalkis, Fisikopoulos, Papachristou, T : 2021])

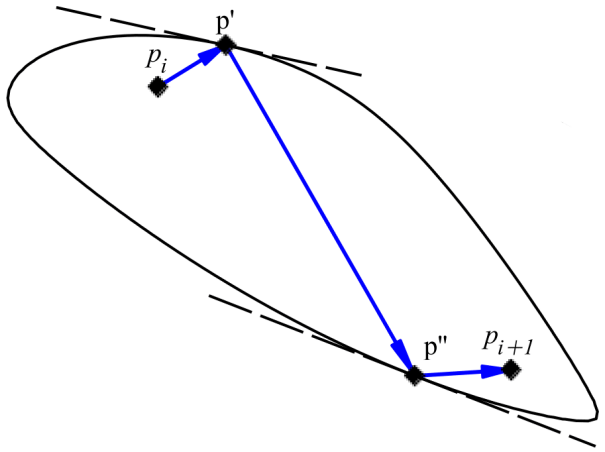
*ReHMC converges to the target distribution when $K$ is a spectrahedron.*

# Outline

# Geometric and algebraic oracles

For (almost) all geometric random walks we need:

- Membership oracle
- Boundary (intersection) oracle
- Reflection oracle



# Membership oracle

MEMBERSHIP$(\boldsymbol{F}, \boldsymbol{p})$: An LMI $\boldsymbol{F}(\boldsymbol{x}) \succeq 0 \Leftrightarrow \boldsymbol{A}_0 + x_1\boldsymbol{A}_1 + \cdots + x_d\boldsymbol{A}_d \succeq 0$ representing a spectrahedron $S$ and a point $\boldsymbol{p} \in \mathbb{R}^d$.

1. $\lambda_{min} \leftarrow$ smallest eigenvalue of $\boldsymbol{F}(\boldsymbol{p})$.
2. **if** $\lambda_{min} \geq 0$ **return** TRUE **else return** FALSE.

# Boundary oracle

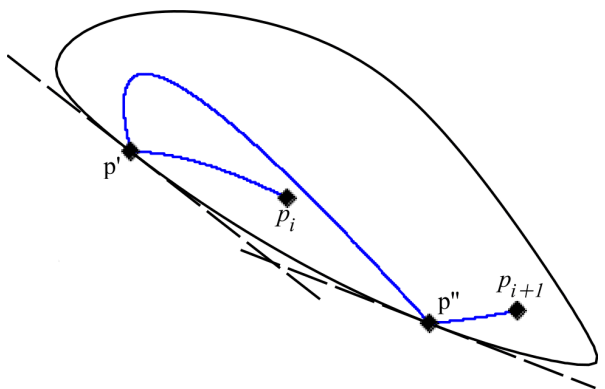
INTERSECTION$(\boldsymbol{F}, \Phi(t))$:
An LMI $\boldsymbol{F}(\boldsymbol{x}) \succeq 0 \Leftrightarrow \boldsymbol{A}_0 + x_1\boldsymbol{A}_1 + \cdots + x_d\boldsymbol{A}_d \succeq 0$ for a spectrahedron $S$, $\Phi : t \mapsto \Phi(t) := (p_1(t), \ldots, p_d(t))$ parameterization of a polynomial curve, where $p_i(t) = \sum_{j=0}^{n_i} p_{i,j} t^j$, and $\Phi(\mathbf{0}) \in S$.

1. Solve the polynomial eigenvalue problem

$$F(\Phi(t))\,\boldsymbol{x} = 0 \Leftrightarrow (\boldsymbol{B}_0 + t\boldsymbol{B}_1 + \cdots + t^d\boldsymbol{B}_d)\boldsymbol{x} = 0,$$

   where $\boldsymbol{B}_k = \sum_{j=1}^{d} p_{j,k}\,\boldsymbol{A}_j$

2. Smallest positive and largest negative eigenvalues $\lambda^-_{max}, \lambda^+_{min}$
3. **return** the boundary points $F(\Phi(\lambda^-_{max}))$ and $F(\Phi(\lambda^+_{min}))$

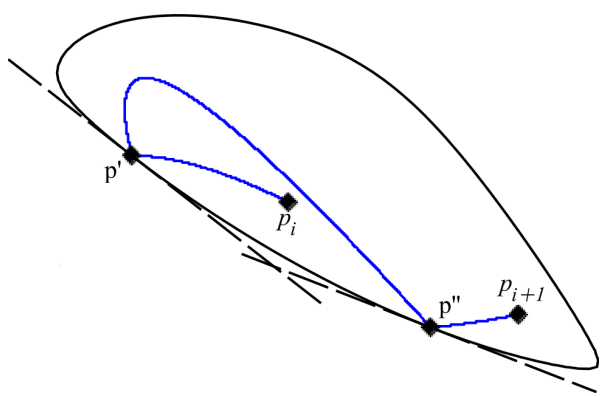


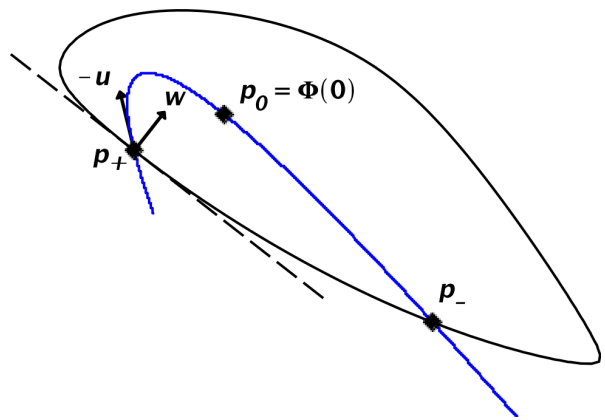
# Reflection oracle

REFLECTION$(\boldsymbol{F}, \Phi(t), \lambda_+)$:
An LMI $\boldsymbol{F}(\boldsymbol{x}) \succeq 0 \Leftrightarrow \boldsymbol{A}_0 + x_1\boldsymbol{A}_1 + \cdots + x_d\boldsymbol{A}_d \succeq 0$ for spectrahedron $S$,

$\Phi(t)$ parameterization of a polynomial curve, $\lambda_+$ s.t. $\Phi(\lambda_+) \in \partial S$

1. Let the boundary point $\boldsymbol{p}_+ = \Phi(\lambda_+)$
2. Let $\boldsymbol{w} = \nabla \det(\boldsymbol{F}(\boldsymbol{p}_+)) = c \cdot (\boldsymbol{s}^\top \boldsymbol{A}_1 \boldsymbol{s}, \cdots, \boldsymbol{s}^\top \boldsymbol{A}_d \boldsymbol{s})$,
   $\boldsymbol{s}$ vector in the kernel of $\boldsymbol{F}(\boldsymbol{p}_+)$
3. **return** the direction of the reflection
   $\boldsymbol{s}_+ \leftarrow \frac{d\Phi}{dt}(t_+) - 2\left\langle \nabla \frac{d\Phi}{dt}(t_+), \boldsymbol{w} \right\rangle \boldsymbol{w}$



# Per-step complexity

| Random walk | per-step Complexity |
|---|---|
| HR | $\mathcal{O}(m^\omega + m\log(1/\epsilon) + dm^2)$ |
| Coordinate HR | $\mathcal{O}(m^\omega + m\log(1/\epsilon) + m^2)$ |
| Billiard walk | $\widetilde{\mathcal{O}}(\rho(m^\omega + m\log(1/\epsilon) + dm^2))$ |
| ReHMC (collocation) | $\widetilde{\mathcal{O}}(\rho((nm)^\omega + mn\log(1/\epsilon) + dnm^2))$ |
| ReHMC (leapfrog) | $\widetilde{\mathcal{O}}(L\rho(m^\omega + m\log(1/\epsilon) + dm^2))$ |

[Chalkis,Fisikopoulos,Repouskos,T: 2019]
[Chalkis,Emiris,Fisikopoulos,Repouskos,T: 2020]

$m$: size of the matrices $\boldsymbol{A}_i$ in LMI
$d$: dimension
$n$: degree of the polynomial curve
$\rho$: number of reflections
$\epsilon$: accuracy to approximate the intersection with the boundary
$\omega$: exponent in the complexity of matrix multiplication

$L$: number of leapfrog steps

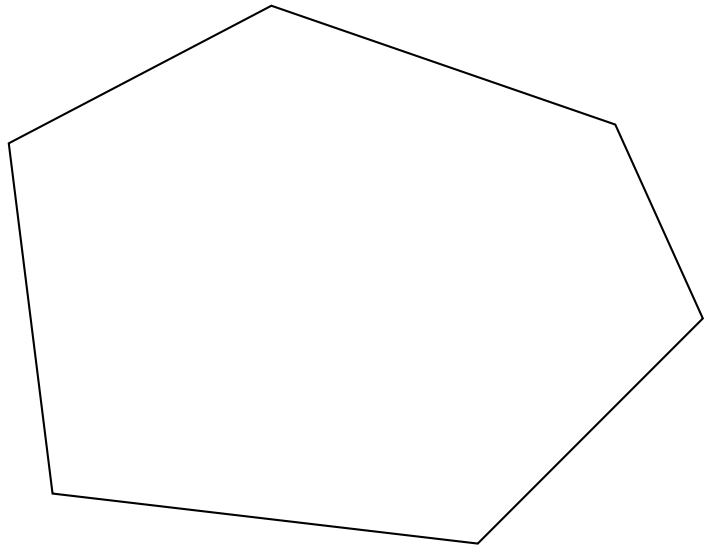
# Outline

# Outline

# Cutting planes

## Dabbene, Shcherbakov, Polyak, 10'

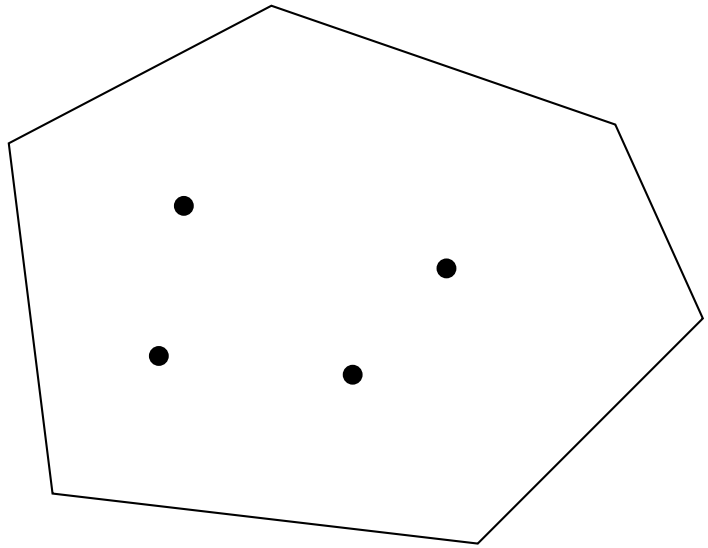
- Input: convex body $K$, objective function $c$.



# Cutting planes

## Dabbene, Shcherbakov, Polyak, 10'

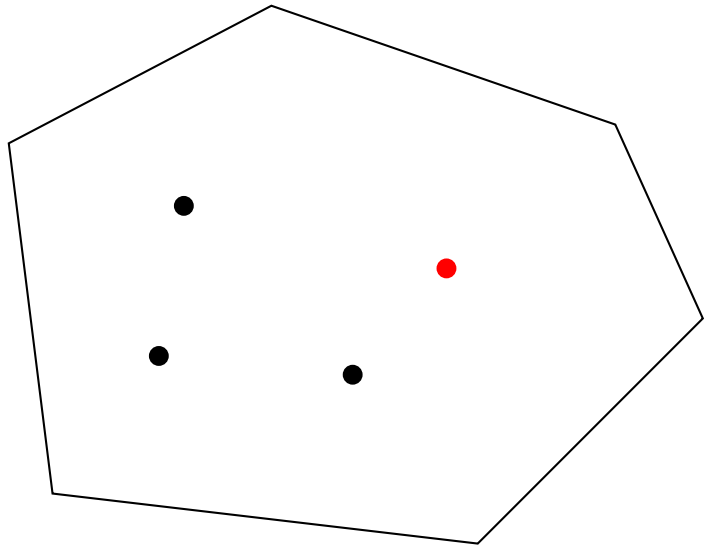
- Input: convex body $K$, objective function $c$.
- Sample $N$ points under the uniform distribution.



# Cutting planes

## Dabbene, Shcherbakov, Polyak, 10'

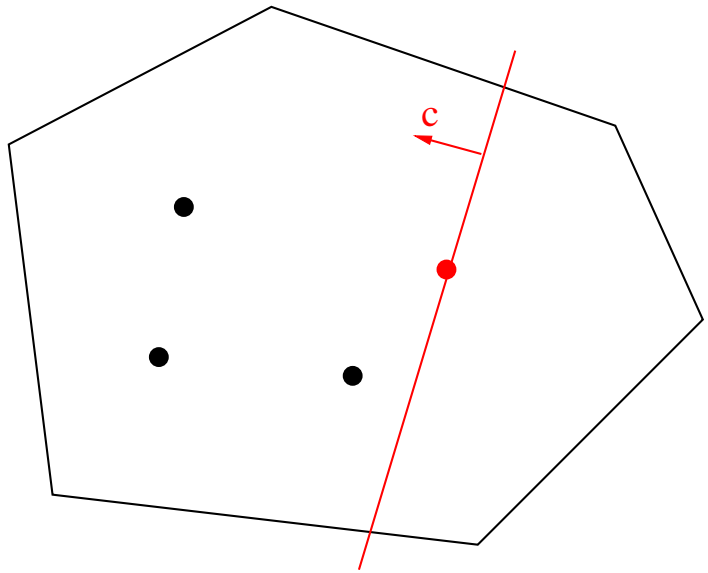
- Input: convex body $K$, objective function $c$.
- Sample $N$ points under the uniform distribution.
- Find the point $x$ minimizing the objective function.



# Cutting planes

## Dabbene, Shcherbakov, Polyak, 10'

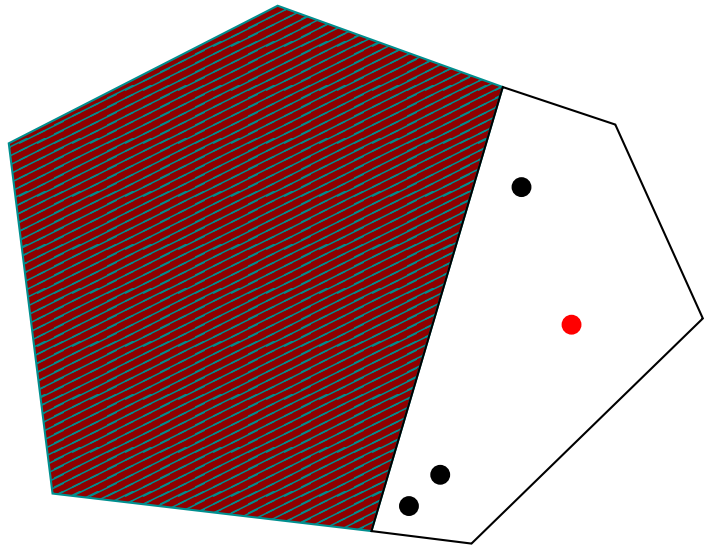
- Input: convex body $K$, objective function $c$.
- Sample $N$ points under the uniform distribution.
- Find the point $x$ minimizing the objective function.
- Cut the convex body at $x$.



# Cutting planes

Dabbene, Shcherbakov, Polyak, 10'

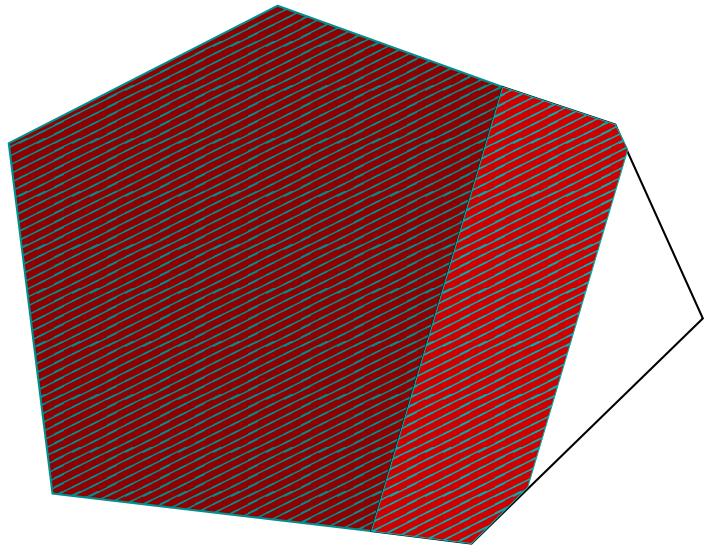
- Input: convex body $K$, objective function $c$.
- Sample $N$ points under the uniform distribution.
- Find the point $x$ minimizing the objective function.
- Cut the convex body at $x$.
- Repeat $I$ times.



# Cutting planes

## Dabbene, Shcherbakov, Polyak, 10'

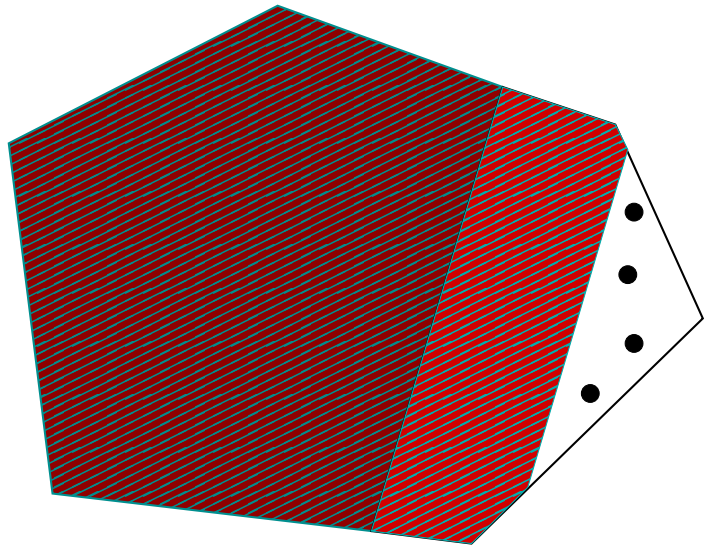
- Input: convex body $K$, objective function $c$.
- Sample $N$ points under the uniform distribution.
- Find the point $x$ minimizing the objective function.
- Cut the convex body at $x$.
- Repeat $I$ times.



# Cutting planes

Dabbene, Shcherbakov, Polyak, 10'

- Input: convex body $K$, objective function $c$.
- Sample $N$ points under the uniform distribution.
- Find the point $x$ minimizing the objective function.
- Cut the convex body at $x$.
- Repeat $I$ times.



# Cutting planes

- Let $rB_d \subseteq K \subseteq RB_d$.
- The expected number of phases s.t. $|f_l - f^*| < \epsilon$ is,

$$l = \left\lceil \frac{1}{\ln(N+1)} d \ln(R/\epsilon) \right\rceil = \widetilde{O}(d)$$

- Total number of uniform points minimized for $N = 1$.
- Total cost,

$$\left\lceil d \ln(R/\epsilon) \right\rceil \times \text{ cost per point}$$

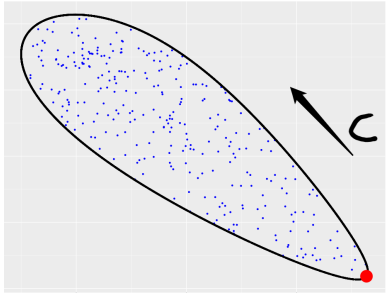
Only Hit&Run has been used up to now
[Bertsimas, Vempala : 2010],[Dabbene, Shcherbakov, Polyak : 2010]
ongoing [Chalkis, Fisikopoulos, Papachristou, T : 2020–]

# Outline

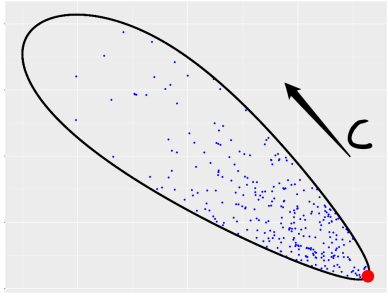
# SDP using Exponential sampling

Problem: Minimize $f(\mathbf{x}) = \boldsymbol{c}^T \boldsymbol{x}$ over a spectrahedron $S$.

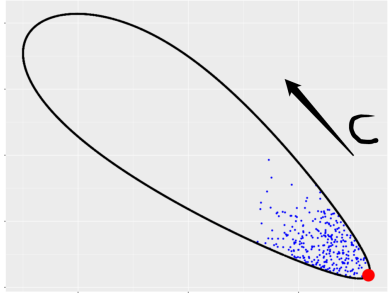
Answer: Sample from $\pi(\boldsymbol{x}) \propto e^{-\boldsymbol{c}^T \boldsymbol{x}/T}$ restricted in $S$, for $T = T_0 > \cdots > T_M$.



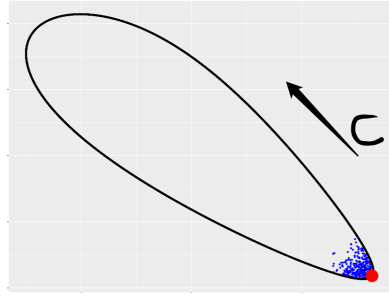

$T_0$




$T_1$




$T_2$

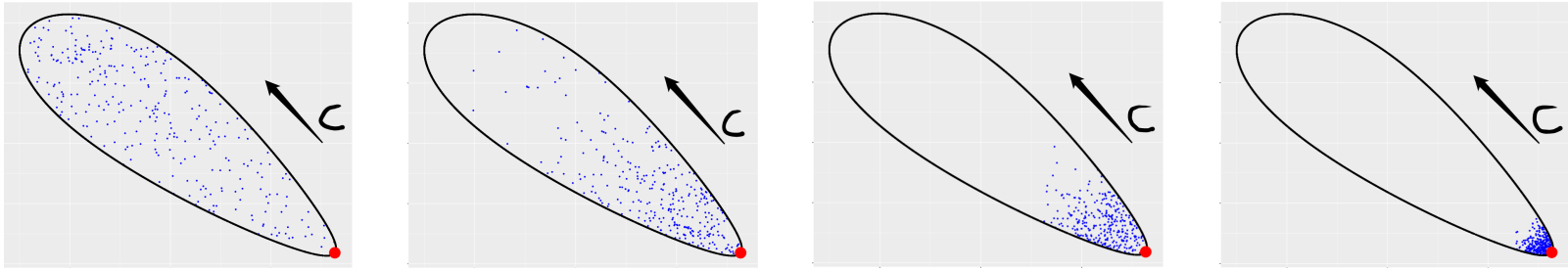



$T_3$

Task: Compute a sequence of $T_i \in \mathbb{R}_+$ of length $M$ s.t. a sample from $\pi_{T_M}$ is close to the optimal solution with high probability.

# Simulated Annealing

## Convergence to the optimal solution




$$\pi_i(\boldsymbol{x}) \propto e^{-\boldsymbol{c}^T \boldsymbol{x}/T_i}$$

- Starting with $T_0 = R$, where $S \subset R\mathcal{B}_d$ (uniform distribution).
- $T_i = T_{i-1}(1 - \frac{1}{\sqrt{d}})$, $i \in [M]$ ($T_{i-1}$ is a warm start for $T_i$).
- $M = \widetilde{O}(\sqrt{d})$ phases to obtain a solution $|f_M - f^*| \leq \epsilon$
- Only Hit-and-Run has been used in previous work [Kalai, Vempala : 2006].

ongoing [Chalkis, Fisikopoulos, Papachristou, T : 2020–]

# Outline

# Volume

## Complexity

Computing the exact volume of $P$,

- is #P-hard for all the representations [DyerFrieze'88]
- is open if both H- and V- representations available
- is APX-hard (oracle model) [Elekes'86]

## Theorem

[Dyer, Frieze, Kannan'91] For any convex body $P$ and any $0 \leq \epsilon,\ \delta \leq 1$, there is a randomized algorithm which computes an estimate $V$ s.t. with probability $1-\delta$ we have $(1-\epsilon)vol(P) \leq V \leq (1+\epsilon)vol(P)$, and the number of oracle calls is $poly(d, 1/\epsilon, log(1/\delta))$.

# Multiphase Monte Carlo

Let a sequence of functions $\{f_0, \ldots, f_m\}$, $f_i : \mathbb{R}^d \to \mathbb{R}$. Then,

$$\text{vol}(P) = \int_P dx = \int_P f_m(x)dx \, \frac{\int_P f_{m-1}(x)dx}{\int_P f_m(x)dx} \cdots \frac{\int_P f_0(x)dx}{\int_P f_1(x)dx} \, \frac{\int_P dx}{\int_P f_0(x)dx}$$

Then select $f_i$ s.t.,

- The number of phases, $m$, is as small as possible.
- Each integral ratio can be efficiently estimated by sampling from $\pi \propto f_i$ restricted to $P$ (using geometric random walks).
- There is a closed formula for $\int_P f_m(x)dx$.

complexity = #phases $\times$ #points per phase $\times$ cost per point

# State-of-the-art

| Authors-Year | Complexity (oracle calls) | $f_i$ | random walk |
|---|---|---|---|
| [Dyer, Frieze, Kannan'91] | $\widetilde{O}(d^{23})$ | Indicator function of a ball | grid walk |
| [Kannan, Lovasz, Simonovits'97] | $\widetilde{O}(d^5)$ | Indicator function of a ball | ball walk |
| [Lovasz, Vempala'03] | $\widetilde{O}(d^4)$ | Exponential | hit-and-run |
| [Cousins, Vempala'15] | $\widetilde{O}(d^3)$ | Spherical Gaussians | ball walk |

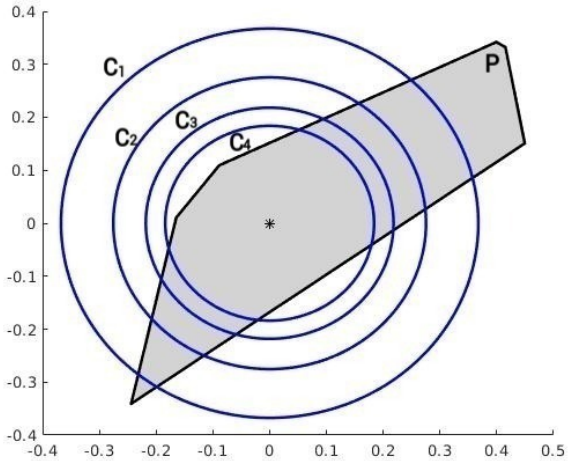
- Can not be implemented as they are due to large constants in the complexity and pessimistic theoretical bounds.

Practical algorithms:

- Follow the theory but make practical adjustments (experimental).
- [Emiris, Fisikopoulos'14] Sequence of balls + coordinate hit-and-run.
- [Cousins, Vempala'16] Spherical Gaussians + hit-and-run

# Multiphase Monte Carlo

- Let $C_m \subseteq \cdots \subseteq C_1$ a sequence of concentric balls intersecting $P$, s.t. $C_m \subseteq P \subseteq C_1$.



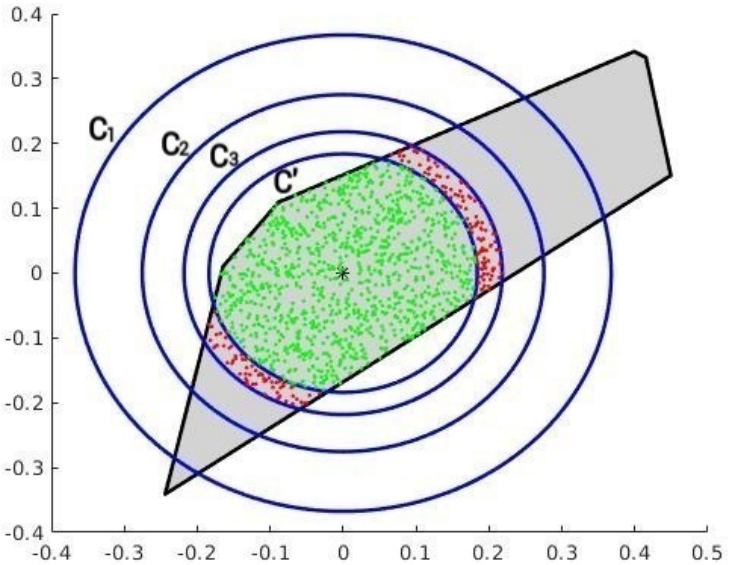

- Construct a sequence of balls intersecting $P$, then:

$$\mathrm{vol}(P) = \mathrm{vol}(P \cap C_m) \frac{\mathrm{vol}(P \cap C_{m-1})}{\mathrm{vol}(P \cap C_m)} \cdots \frac{\mathrm{vol}(P \cap C_1)}{\mathrm{vol}(P \cap C_2)} \frac{\mathrm{vol}(P)}{\mathrm{vol}(P \cap C_1)}$$

$$m = \lceil d \lg \frac{R}{r} \rceil$$

# Ratio estimation

- Estimate $r_i = \frac{\text{vol}(P \cap C_{i+1})}{\text{vol}(P \cap C_i)}$ within some target relative error $\epsilon_i$.
- Sample $N$ uniform points from $P_i = C_i \cap P$ and count points in $P_{i+1} = C_{i+1} \cap P \subseteq P_i$.




- Keep each ratio bounded, then $N = O(1/\epsilon_i^2)$ points suffices.

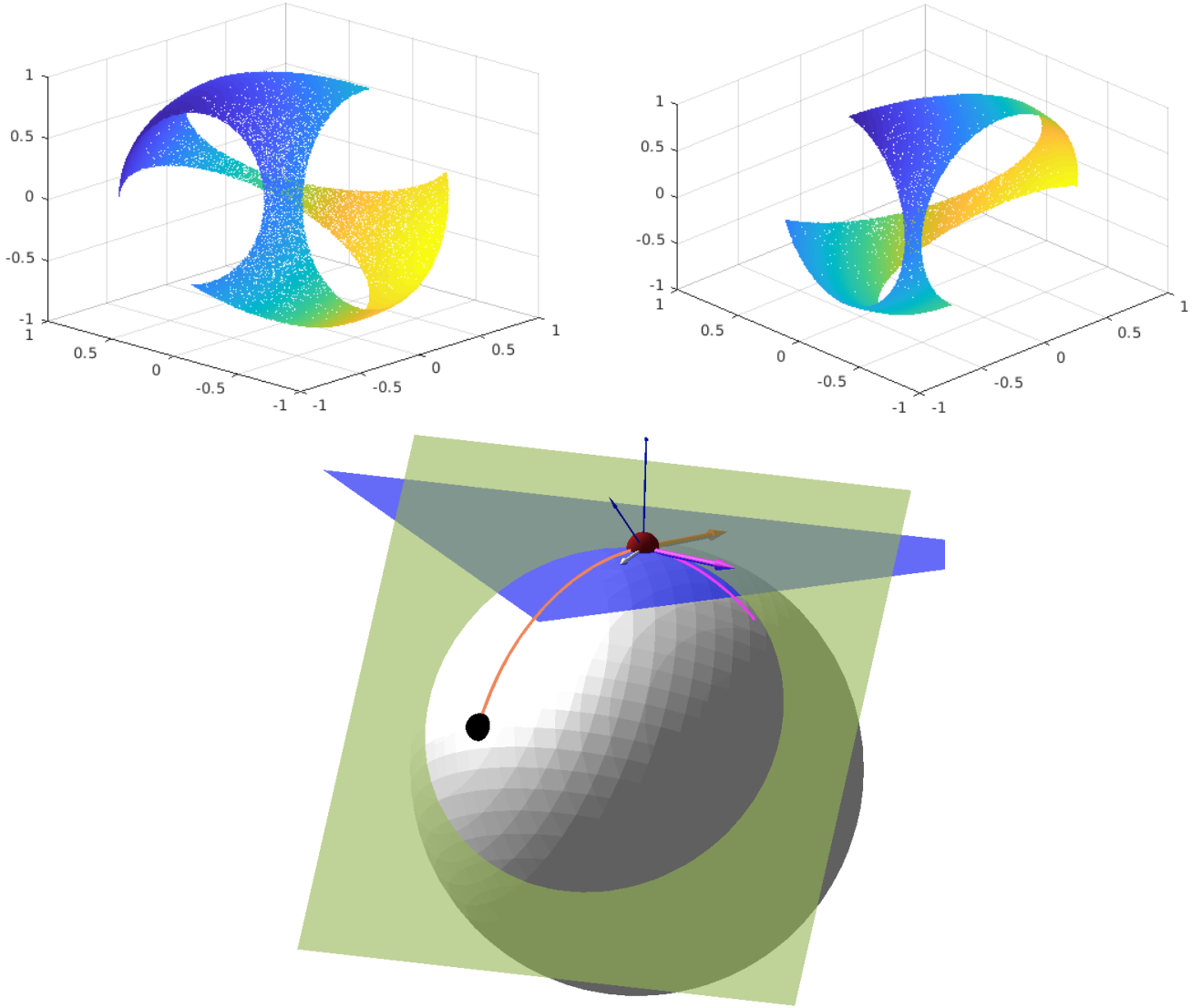
# Does this approach work?

| *S-n-m* | $\mu \pm t_{\alpha,\nu-1}\frac{s}{\sqrt{\nu}}$ | *Points* | *Time (sec)* | *error* |
|---|---|---|---|---|
| *S*-40-40 | $(1.34 \pm 0.12)$e-06 | 9975.2 | 6.7 | ?? |
| *S*-60-60 | $(1.23 \pm 0.11)$e-20 | 20370.9 | 28.5 | ?? |
| *S*-80-80 | $(4.24 \pm 0.26)$e-33 | 31539.1 | 124.4 | ?? |
| *S*-100-100 | $(1.21 \pm 0.10)$e-51 | 52962.7 | 362.3 | ?? |
| **S*-28-8 | $14.31 \pm 0.64$ | 4547.4 | 10.2 | 0.05 |
| **S*-45-10 | $0.6334 \pm 0.03$ | 19558.1 | 56.2 | 0.07 |
| **S*-66-12 | $(1.73 \pm 0.034)$e-03 | 1.01e+05 | 324.2 | 0.07 |

Table: $m$ is the matrix dimension in LMI and $n$ the ambient dimension. The spectrahedra marked with "*" are elliptopes, $\mu$ stands for the average volume and $s$ for the standard deviation. We give a confidence interval with level of confidence $\alpha = 0.05$, while $t_{\alpha,\nu-1}$ is the critical value of student's distribution with $\nu - 1$ degrees of freedom. Error parameter to $e = 0.1$.

# Outline

# Volatility detection




[Bachelard, Chalkis, Fisikopoulos, T : 2022]

# (IMO) Some very interesting questions

- What is the arithmetic/bit complexity of producing one sample in a polytope/spectrahedron $\epsilon$ close to the uniform distribution?
  What about any log-concave distribution?
- What is the arithmetic/bit complexity of computing the volume of a polytope?
- What is the arithmetic/bit complecity of computing the volume of a spectrahedron?
- What is the arithmetic/bit complexity of putting a polytope/spectrahedron in almost isotropic position?
  What do we mean by almost?
- What is arithmetic/boolean complexity of LP and SDP
  using sampling and cutting planes?

# GeomScale org

https://geomscale.github.io


– GeomScale/volesti
volume approximation & sampling
from convex bodies


– GeomScale/dingo
analyze metabolic networks with
MCMC sampling

Co-founders: Tolis Chalkis & Vissarion Fisikopoulos & E.T.

---

NumFOCUS Affiliated Project.

Support from an open community.

More than 15 000 lines of code.

**Thank you!**