LECTURE 8

INTRODUCTION TO PARALLEL ANNEALING

Saifuddin Syed

PROBLEMS WITH SERIAL MCMC

PROBLEMS WITH SERIAL MCMC

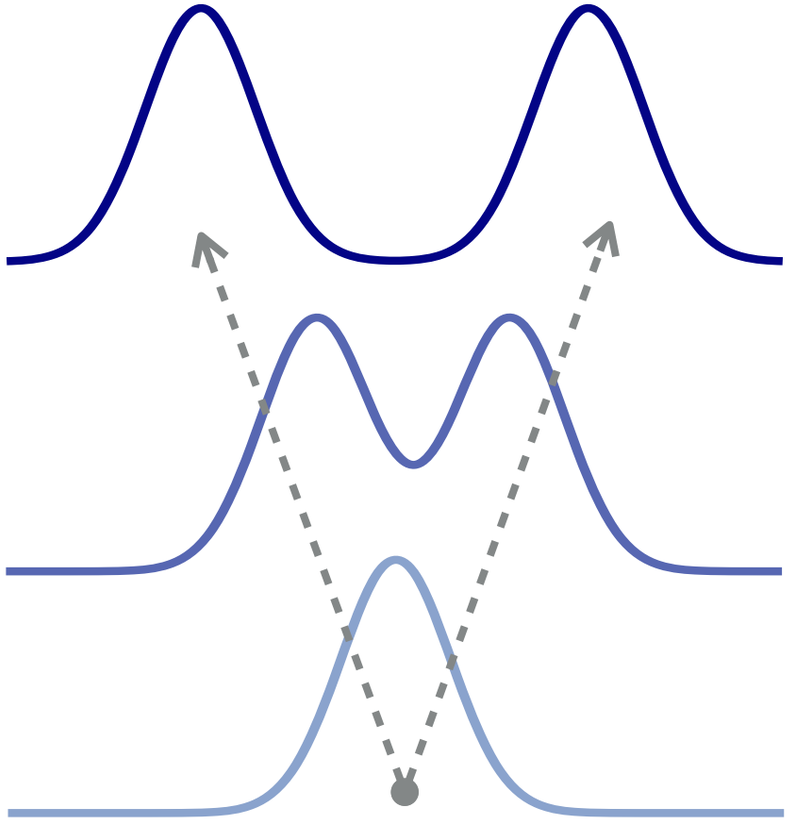
- **Hardware:** Clock speeds are stagnant, and modern computation is distributed

PROBLEMS WITH SERIAL MCMC

- **Hardware:** Clock speeds are stagnant, and modern computation is distributed
- **Algorithmic:** Single chain tries to simultaneously efficiently explore & be accurate

PROBLEMS WITH SERIAL MCMC

- **Hardware:** Clock speeds are stagnant, and modern computation is distributed
- **Algorithmic:** Single chain tries to simultaneously efficiently explore & be accurate
- **Solution:** Build an interacting particle system
  - Delegate exploration to tractable **reference** chain
  - Communicate to **target** through the **interpolating distributions**

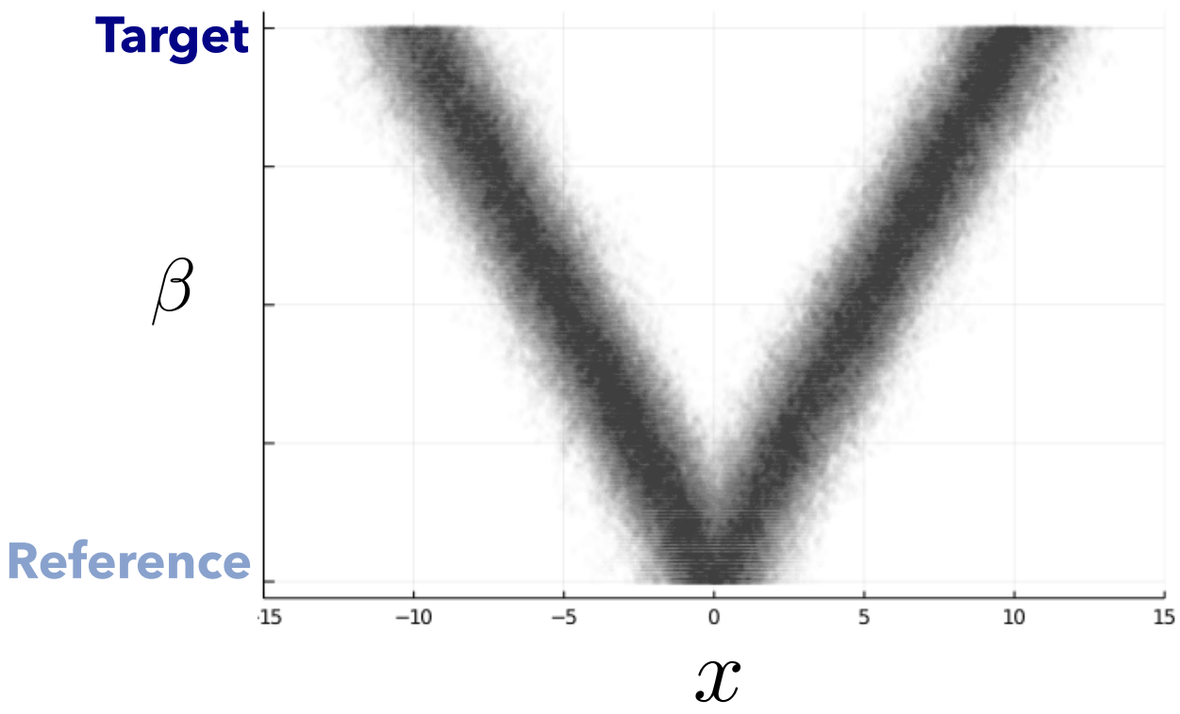


ANNEALING ALGORITHMS

- Annealing is **not** an algorithm but a platform to build algorithms on top of

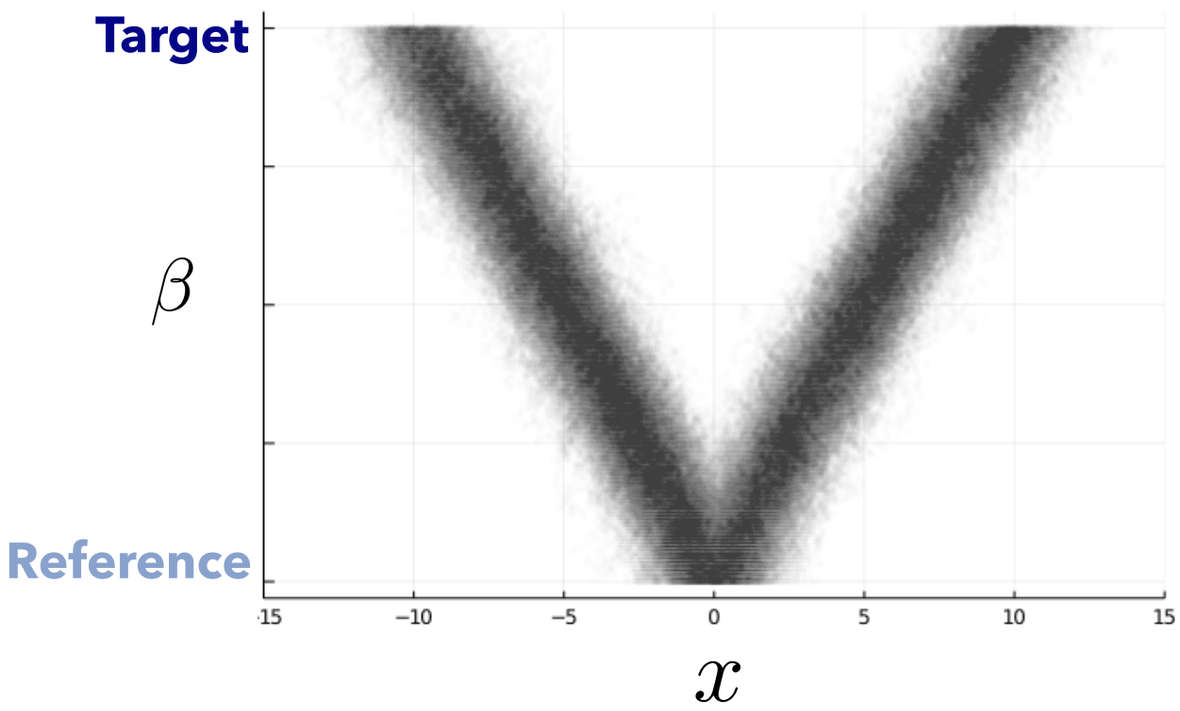
ANNEALING ALGORITHMS

- Annealing is **not** an algorithm but a platform to build algorithms on top of



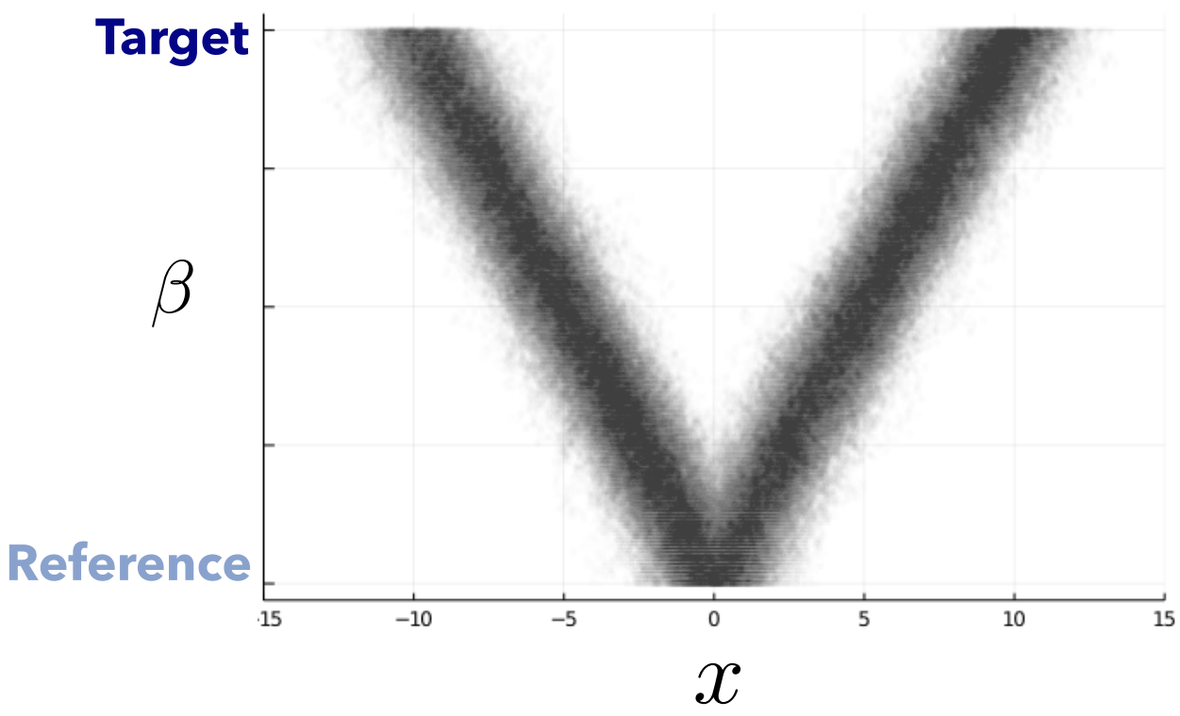
ANNEALING ALGORITHMS

- Annealing is **not** an algorithm but a platform to build algorithms on top of
- **Annealing algorithms** draw inference from the entire path not just the target



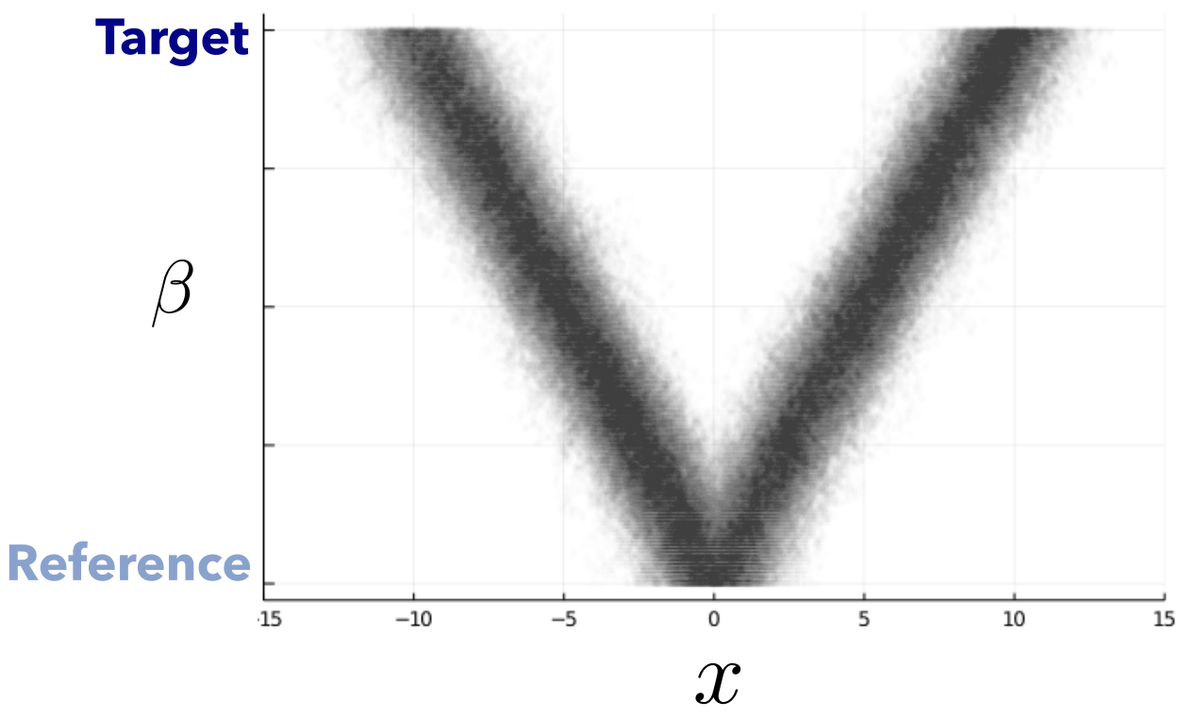
ANNEALING ALGORITHMS

- Annealing is **not** an algorithm but a platform to build algorithms on top of
- **Annealing algorithms** draw inference from the entire path not just the target
- Annealing algorithms are meta algorithms:



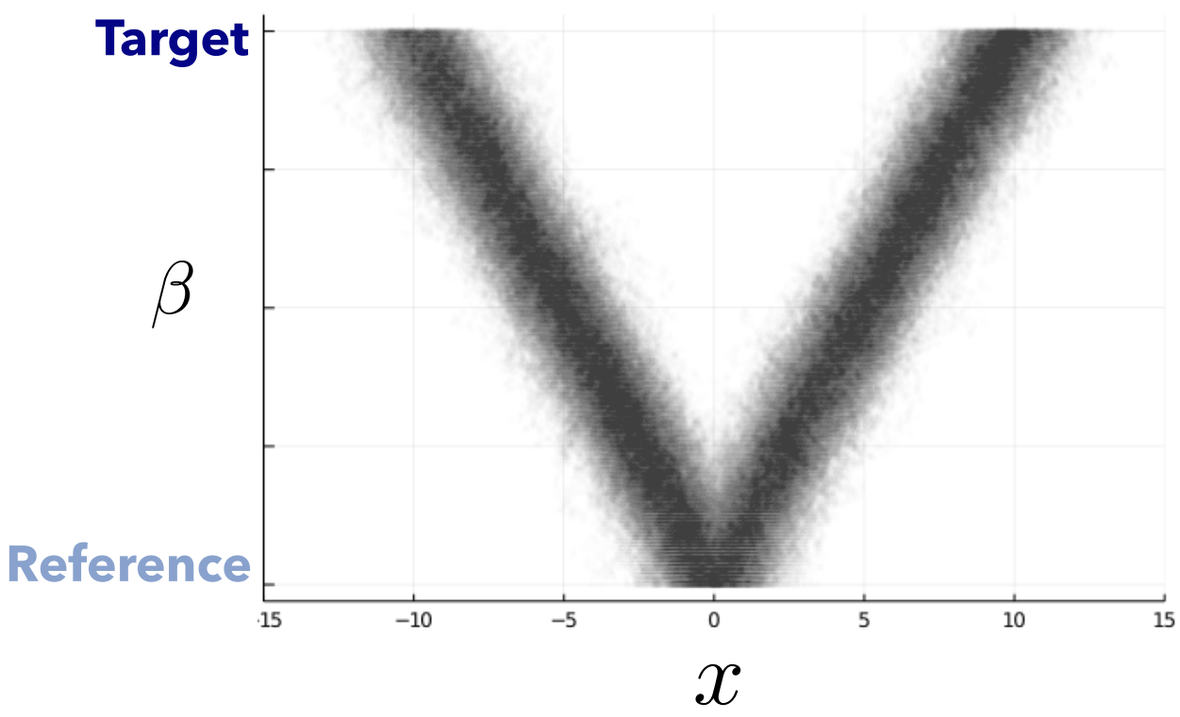
ANNEALING ALGORITHMS

- Annealing is **not** an algorithm but a platform to build algorithms on top of
- **Annealing algorithms** draw inference from the entire path not just the target
- Annealing algorithms are meta algorithms:
  - **Input:** An annealing path π_β + a local efficient inference algorithm for each π_β



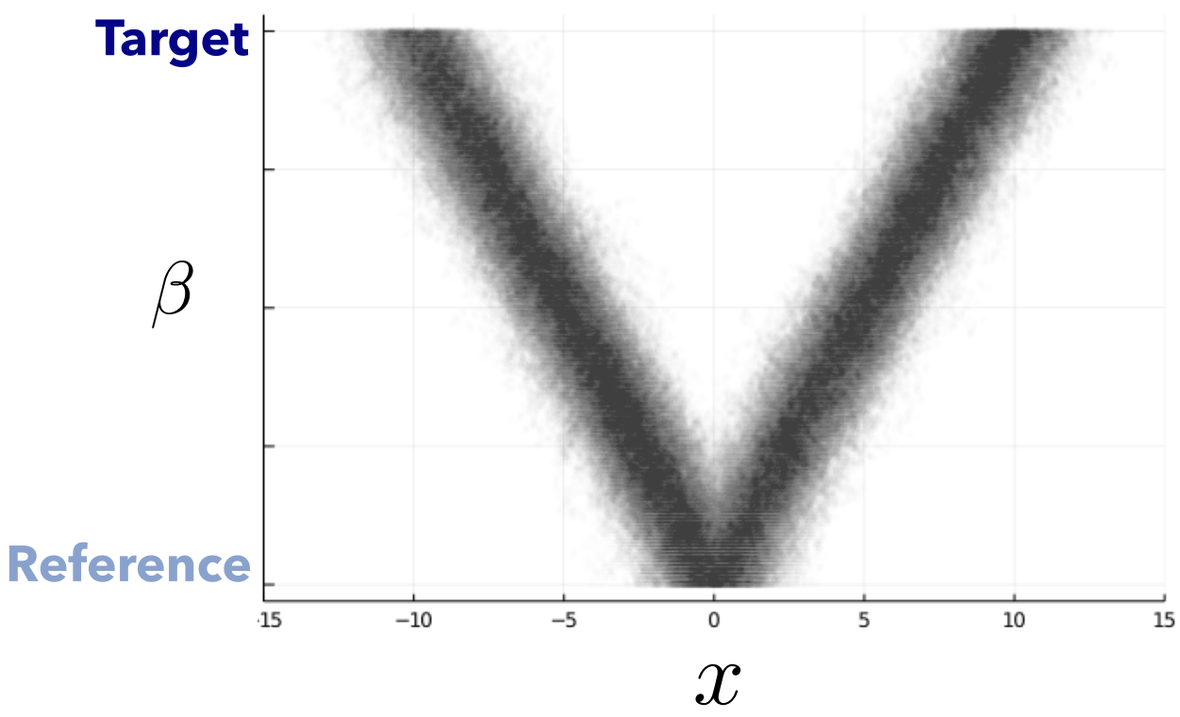
ANNEALING ALGORITHMS

- Annealing is **not** an algorithm but a platform to build algorithms on top of
- **Annealing algorithms** draw inference from the entire path not just the target
- Annealing algorithms are meta algorithms:
  - **Input:** An annealing path π_β + a local efficient inference algorithm for each π_β
  - **Output:** a globally efficient inference algorithm for the path $\beta \mapsto \pi_\beta$



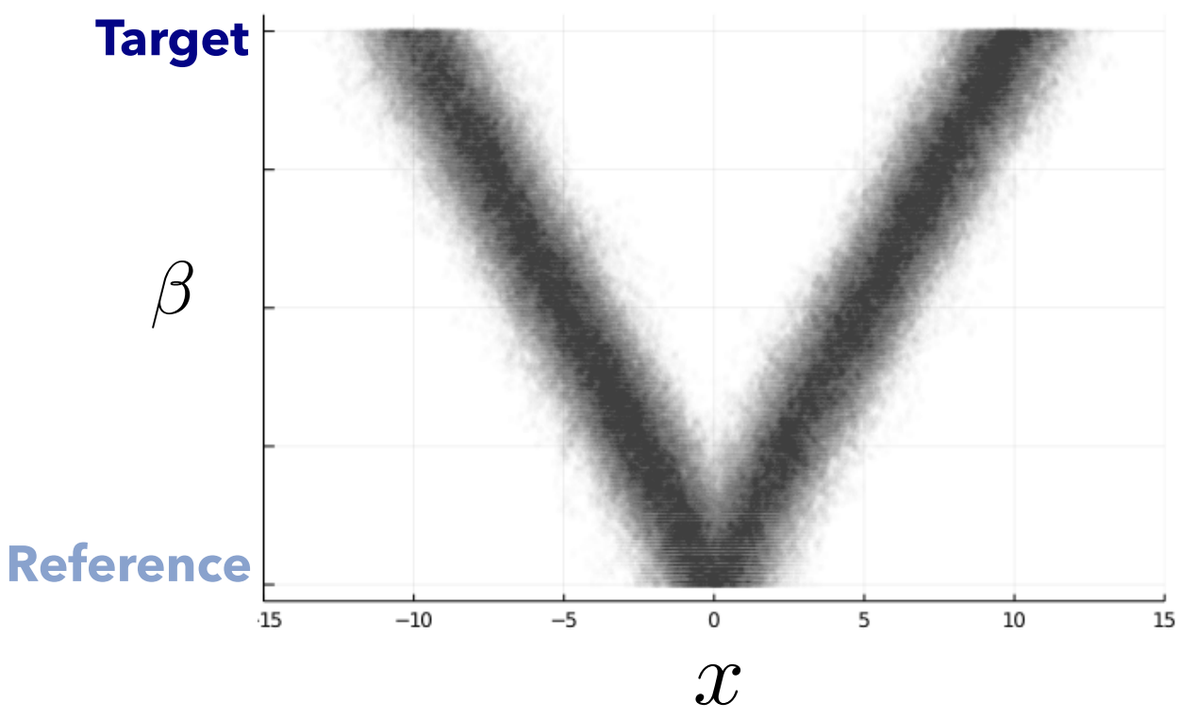
ANNEALING ALGORITHMS

- Annealing is **not** an algorithm but a platform to build algorithms on top of
- **Annealing algorithms** draw inference from the entire path not just the target
- Annealing algorithms are meta algorithms:
  - **Input:** An annealing path π_β + a local efficient inference algorithm for each π_β
  - **Output:** a globally efficient inference algorithm for the path $\beta \mapsto \pi_\beta$
- They achieve state-of-the-art performance for hard mult-modal distributions



ANNEALING ALGORITHMS

- Annealing is **not** an algorithm but a platform to build algorithms on top of
- **Annealing algorithms** draw inference from the entire path not just the target
- Annealing algorithms are meta algorithms:
  - **Input:** An annealing path π_β + a local efficient inference algorithm for each π_β
  - **Output:** a globally efficient inference algorithm for the path $\beta \mapsto \pi_\beta$
- They achieve state-of-the-art performance for hard mult-modal distributions
- Can use the reference to assess the quality and ensure robustness



INGREDIENT FOR AN ANNEALING ALGORITHM

INGREDIENT FOR AN ANNEALING ALGORITHM

- **Target** distribtution $\pi(\mathrm{d}x) = \pi(x)\mathrm{d}x$ supported $\mathbb{X}$ such that:

INGREDIENT FOR AN ANNEALING ALGORITHM

- **Target** distribtution $\pi(\mathrm{d}x) = \pi(x)\mathrm{d}x$ supported $\mathbb{X}$ such that:
  - We we can evaluate the un-normalised density $\gamma(x) \propto \pi(x)$ for all x

INGREDIENT FOR AN ANNEALING ALGORITHM

- **Target** distribtution $\pi(\mathrm{d}x) = \pi(x)\mathrm{d}x$ supported $\mathbb{X}$ such that:
  - We we can evaluate the un-normalised density $\gamma(x) \propto \pi(x)$ for all x
- **Reference** distribution $\eta(\mathrm{d}x) = \eta(x)\mathrm{d}x$ supported on $\mathbb{X}$ such that:

INGREDIENT FOR AN ANNEALING ALGORITHM

- **Target** distribtution $\pi(\mathrm{d}x) = \pi(x)\mathrm{d}x$ supported $\mathbb{X}$ such that:
  - We we can evaluate the un-normalised density $\gamma(x) \propto \pi(x)$ for all x
- **Reference** distribution $\eta(\mathrm{d}x) = \eta(x)\mathrm{d}x$ supported on $\mathbb{X}$ such that:
  - We can efficiently sample and evaluate the normalised density $\eta(x)$ for all $x \in \mathbb{X}$

INGREDIENT FOR AN ANNEALING ALGORITHM

- **Target** distribtution $\pi(\mathrm{d}x) = \pi(x)\mathrm{d}x$ supported $\mathbb{X}$ such that:
  - We we can evaluate the un-normalised density $\gamma(x) \propto \pi(x)$ for all x
- **Reference** distribution $\eta(\mathrm{d}x) = \eta(x)\mathrm{d}x$ supported on $\mathbb{X}$ such that:
  - We can efficiently sample and evaluate the normalised density $\eta(x)$ for all $x \in \mathbb{X}$
- **Annealing path** of distribution $\pi_\beta(\mathrm{d}x) = \pi_\beta(x)\mathrm{d}x$ supported on $\mathbb{X}$ such that

INGREDIENT FOR AN ANNEALING ALGORITHM

- **Target** distribtution $\pi(\mathrm{d}x) = \pi(x)\mathrm{d}x$ supported $\mathbb{X}$ such that:
  - We we can evaluate the un-normalised density $\gamma(x) \propto \pi(x)$ for all x
- **Reference** distribution $\eta(\mathrm{d}x) = \eta(x)\mathrm{d}x$ supported on $\mathbb{X}$ such that:
  - We can efficiently sample and evaluate the normalised density $\eta(x)$ for all $x \in \mathbb{X}$
- **Annealing path** of distribution $\pi_\beta(\mathrm{d}x) = \pi_\beta(x)\mathrm{d}x$ supported on $\mathbb{X}$ such that
  - The path $\beta \mapsto \pi_\beta$ interpolating between the reference and target $\pi_0 = \eta$ and $\pi_1 = \pi$

INGREDIENT FOR AN ANNEALING ALGORITHM

- **Target** distribtution $\pi(\mathrm{d}x) = \pi(x)\mathrm{d}x$ supported $\mathbb{X}$ such that:
  - We we can evaluate the un-normalised density $\gamma(x) \propto \pi(x)$ for all x
- **Reference** distribution $\eta(\mathrm{d}x) = \eta(x)\mathrm{d}x$ supported on $\mathbb{X}$ such that:
  - We can efficiently sample and evaluate the normalised density $\eta(x)$ for all $x \in \mathbb{X}$
- **Annealing path** of distribution $\pi_\beta(\mathrm{d}x) = \pi_\beta(x)\mathrm{d}x$ supported on $\mathbb{X}$ such that
  - The path $\beta \mapsto \pi_\beta$ interpolating between the reference and target $\pi_0 = \eta$ and $\pi_1 = \pi$
  - For each β we we can evaulate the un-normalised density $\gamma_\beta(x) \propto \pi_\beta(x)$

INGREDIENT FOR AN ANNEALING ALGORITHM

- **Target** distribtution $\pi(\mathrm{d}x) = \pi(x)\mathrm{d}x$ supported $\mathbb{X}$ such that:
  - We we can evaluate the un-normalised density $\gamma(x) \propto \pi(x)$ for all x
- **Reference** distribution $\eta(\mathrm{d}x) = \eta(x)\mathrm{d}x$ supported on $\mathbb{X}$ such that:
  - We can efficiently sample and evaluate the normalised density $\eta(x)$ for all $x \in \mathbb{X}$
- **Annealing path** of distribution $\pi_\beta(\mathrm{d}x) = \pi_\beta(x)\mathrm{d}x$ supported on $\mathbb{X}$ such that
  - The path $\beta \mapsto \pi_\beta$ interpolating between the reference and target $\pi_0 = \eta$ and $\pi_1 = \pi$
  - For each β we we can evaulate the un-normalised density $\gamma_\beta(x) \propto \pi_\beta(x)$
- **Local exploration** Markov kernels $K_\beta(x, \mathrm{d}x')$ such that K_β is π_β-invariant for each β, i.e.

INGREDIENT FOR AN ANNEALING ALGORITHM

- **Target** distribtution $\pi(\mathrm{d}x) = \pi(x)\mathrm{d}x$ supported $\mathbb{X}$ such that:
  - We we can evaluate the un-normalised density $\gamma(x) \propto \pi(x)$ for all x
- **Reference** distribution $\eta(\mathrm{d}x) = \eta(x)\mathrm{d}x$ supported on $\mathbb{X}$ such that:
  - We can efficiently sample and evaluate the normalised density $\eta(x)$ for all $x \in \mathbb{X}$
- **Annealing path** of distribution $\pi_\beta(\mathrm{d}x) = \pi_\beta(x)\mathrm{d}x$ supported on $\mathbb{X}$ such that
  - The path $\beta \mapsto \pi_\beta$ interpolating between the reference and target $\pi_0 = \eta$ and $\pi_1 = \pi$
  - For each β we we can evaulate the un-normalised density $\gamma_\beta(x) \propto \pi_\beta(x)$
- **Local exploration** Markov kernels $K_\beta(x, \mathrm{d}x')$ such that K_β is π_β-invariant for each β, i.e.
  - We can efficiently sample from $K_\beta(x, \mathrm{d}x')$ for all $\beta \in [0,1]$ and $x \in \mathbb{X}$

INGREDIENT FOR AN ANNEALING ALGORITHM

- **Target** distribtution $\pi(\mathrm{d}x) = \pi(x)\mathrm{d}x$ supported $\mathbb{X}$ such that:
  - We we can evaluate the un-normalised density $\gamma(x) \propto \pi(x)$ for all x
- **Reference** distribution $\eta(\mathrm{d}x) = \eta(x)\mathrm{d}x$ supported on $\mathbb{X}$ such that:
  - We can efficiently sample and evaluate the normalised density $\eta(x)$ for all $x \in \mathbb{X}$
- **Annealing path** of distribution $\pi_\beta(\mathrm{d}x) = \pi_\beta(x)\mathrm{d}x$ supported on $\mathbb{X}$ such that
  - The path $\beta \mapsto \pi_\beta$ interpolating between the reference and target $\pi_0 = \eta$ and $\pi_1 = \pi$
  - For each β we we can evaulate the un-normalised density $\gamma_\beta(x) \propto \pi_\beta(x)$
- **Local exploration** Markov kernels $K_\beta(x, \mathrm{d}x')$ such that K_β is π_β-invariant for each β, i.e.
  - We can efficiently sample from $K_\beta(x, \mathrm{d}x')$ for all $\beta \in [0,1]$ and $x \in \mathbb{X}$
  - At $\beta = 0$ we have $K_\beta(x, \mathrm{d}x') = \eta(\mathrm{d}x')$ is the independent kernel for η

INGREDIENT FOR AN ANNEALING ALGORITHM

- **Target** distribtution $\pi(\mathrm{d}x) = \pi(x)\mathrm{d}x$ supported $\mathbb{X}$ such that:
  - We we can evaluate the un-normalised density $\gamma(x) \propto \pi(x)$ for all x
- **Reference** distribution $\eta(\mathrm{d}x) = \eta(x)\mathrm{d}x$ supported on $\mathbb{X}$ such that:
  - We can efficiently sample and evaluate the normalised density $\eta(x)$ for all $x \in \mathbb{X}$
- **Annealing path** of distribution $\pi_\beta(\mathrm{d}x) = \pi_\beta(x)\mathrm{d}x$ supported on $\mathbb{X}$ such that
  - The path $\beta \mapsto \pi_\beta$ interpolating between the reference and target $\pi_0 = \eta$ and $\pi_1 = \pi$
  - For each β we we can evaulate the un-normalised density $\gamma_\beta(x) \propto \pi_\beta(x)$
- **Local exploration** Markov kernels $K_\beta(x, \mathrm{d}x')$ such that K_β is π_β-invariant for each β, i.e.
  - We can efficiently sample from $K_\beta(x, \mathrm{d}x')$ for all $\beta \in [0,1]$ and $x \in \mathbb{X}$
  - At $\beta = 0$ we have $K_\beta(x, \mathrm{d}x') = \eta(\mathrm{d}x')$ is the independent kernel for η
- **Annealing schedule** specifying how to discretise the annealing path

INGREDIENT FOR AN ANNEALING ALGORITHM

- **Target** distribtution $\pi(\mathrm{d}x) = \pi(x)\mathrm{d}x$ supported $\mathbb{X}$ such that:
  - We we can evaluate the un-normalised density $\gamma(x) \propto \pi(x)$ for all x
- **Reference** distribution $\eta(\mathrm{d}x) = \eta(x)\mathrm{d}x$ supported on $\mathbb{X}$ such that:
  - We can efficiently sample and evaluate the normalised density $\eta(x)$ for all $x \in \mathbb{X}$
- **Annealing path** of distribution $\pi_\beta(\mathrm{d}x) = \pi_\beta(x)\mathrm{d}x$ supported on $\mathbb{X}$ such that
  - The path $\beta \mapsto \pi_\beta$ interpolating between the reference and target $\pi_0 = \eta$ and $\pi_1 = \pi$
  - For each β we we can evaulate the un-normalised density $\gamma_\beta(x) \propto \pi_\beta(x)$
- **Local exploration** Markov kernels $K_\beta(x, \mathrm{d}x')$ such that K_β is π_β-invariant for each β, i.e.
  - We can efficiently sample from $K_\beta(x, \mathrm{d}x')$ for all $\beta \in [0,1]$ and $x \in \mathbb{X}$
  - At $\beta = 0$ we have $K_\beta(x, \mathrm{d}x') = \eta(\mathrm{d}x')$ is the independent kernel for η
- **Annealing schedule** specifying how to discretise the annealing path

$$0 = \beta_0 < \beta_1 < \cdots < \beta_N = 1$$

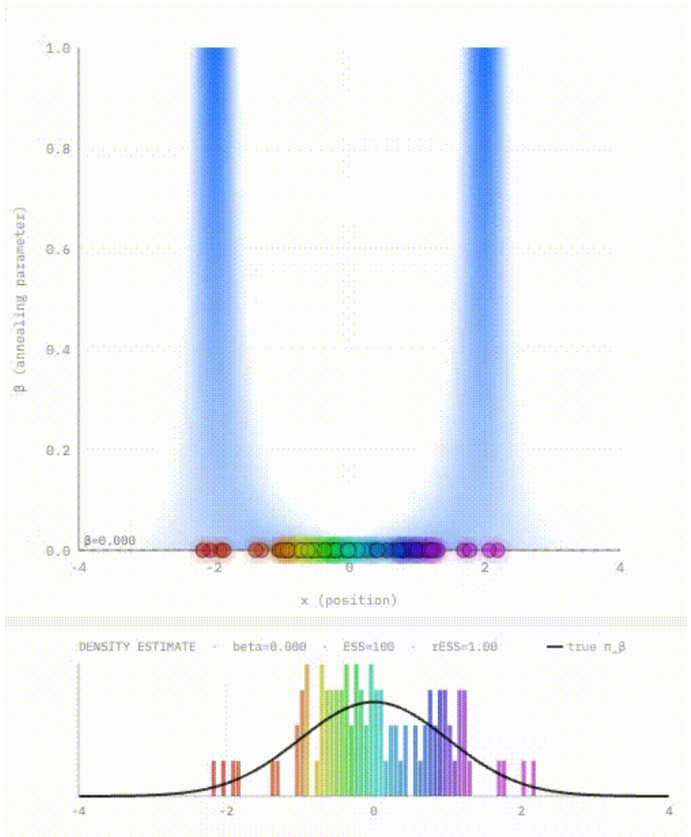
SEQUENTIAL VERSUS PARALLEL ANNEALING

- We simultaneously evolve a system of particles to approximation the annealing path

SEQUENTIAL VERSUS PARALLEL ANNEALING

- We simultaneously evolve a system of particles to approximation the annealing path

Sequential Annealing

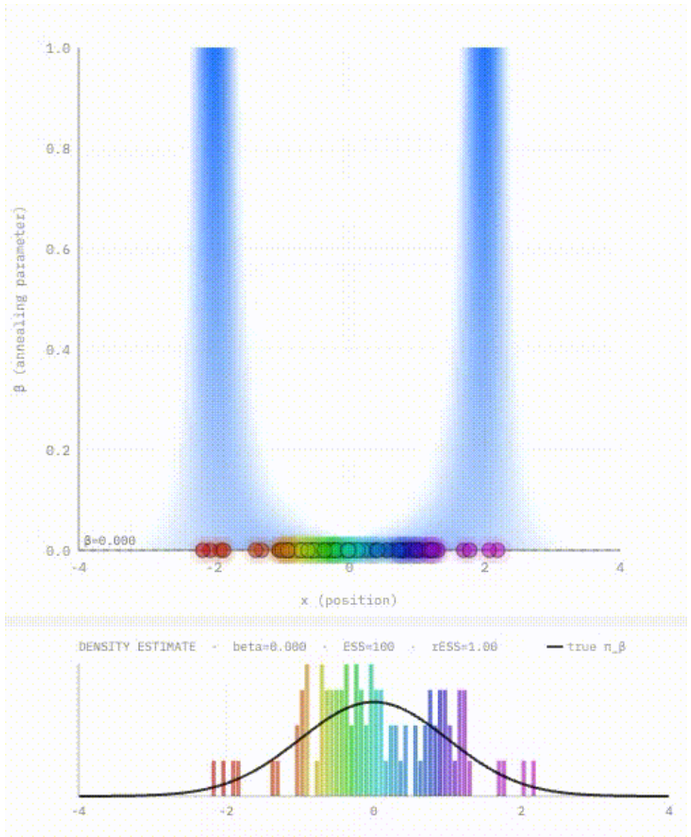


SEQUENTIAL VERSUS PARALLEL ANNEALING

- We simultaneously evolve a system of particles to approximation the annealing path

Sequential Annealing

- Construct path sequentially

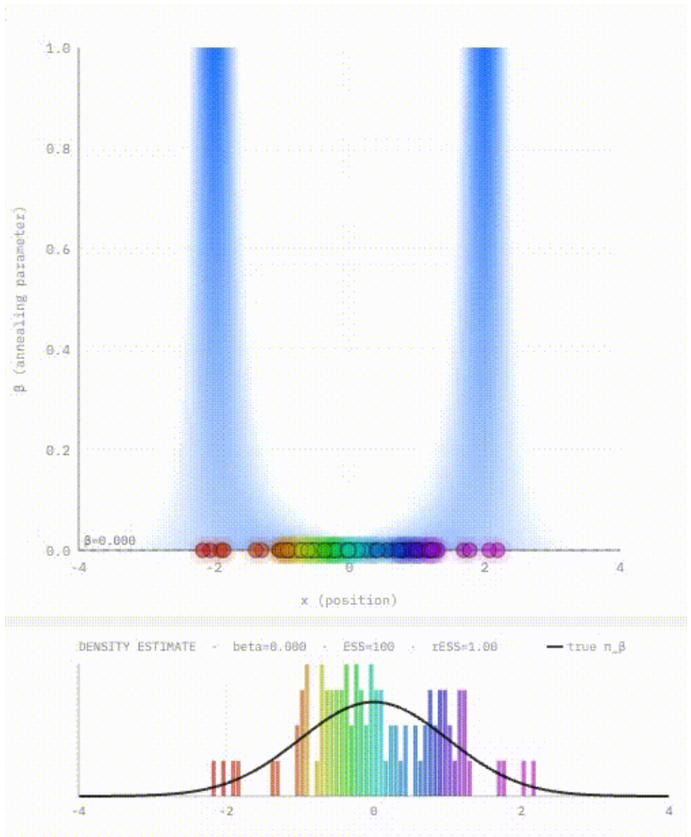


SEQUENTIAL VERSUS PARALLEL ANNEALING

- We simultaneously evolve a system of particles to approximation the annealing path

Sequential Annealing

- Construct path sequentially
- Samples generated in parallel

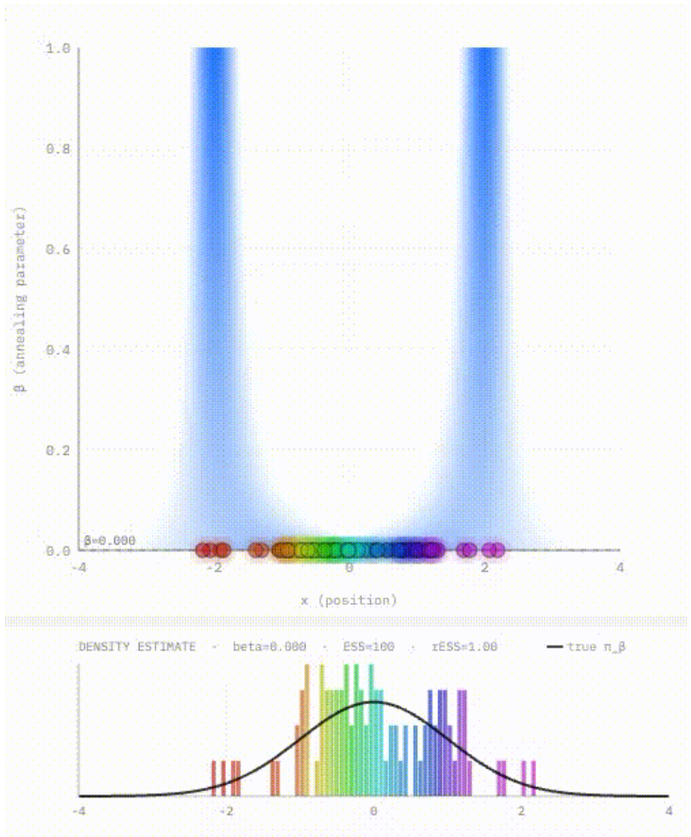


SEQUENTIAL VERSUS PARALLEL ANNEALING

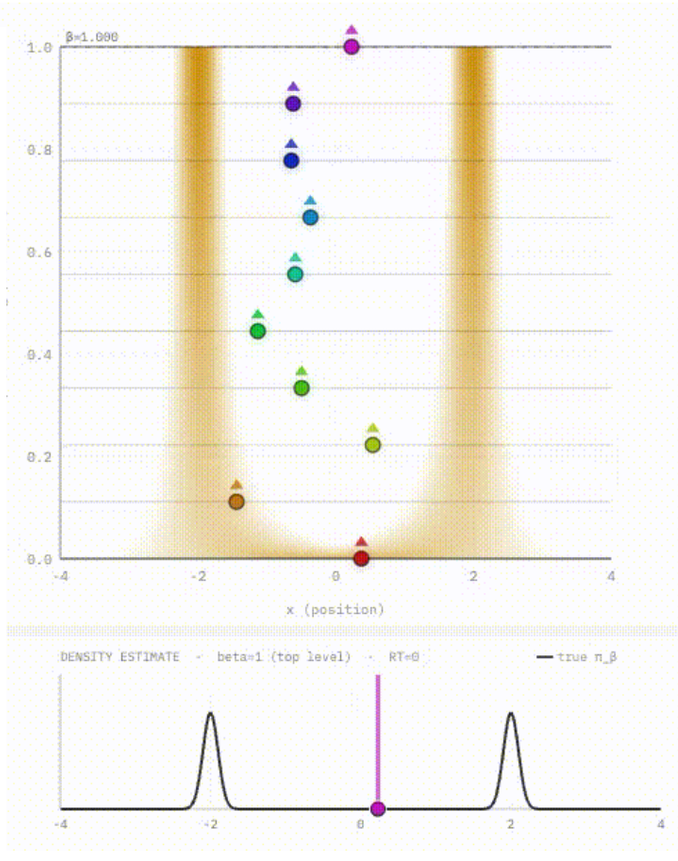
- We simultaneously evolve a system of particles to approximation the annealing path

Sequential Annealing

- Construct path sequentially
- Samples generated in parallel



Parallel Annealing



SEQUENTIAL VERSUS PARALLEL ANNEALING

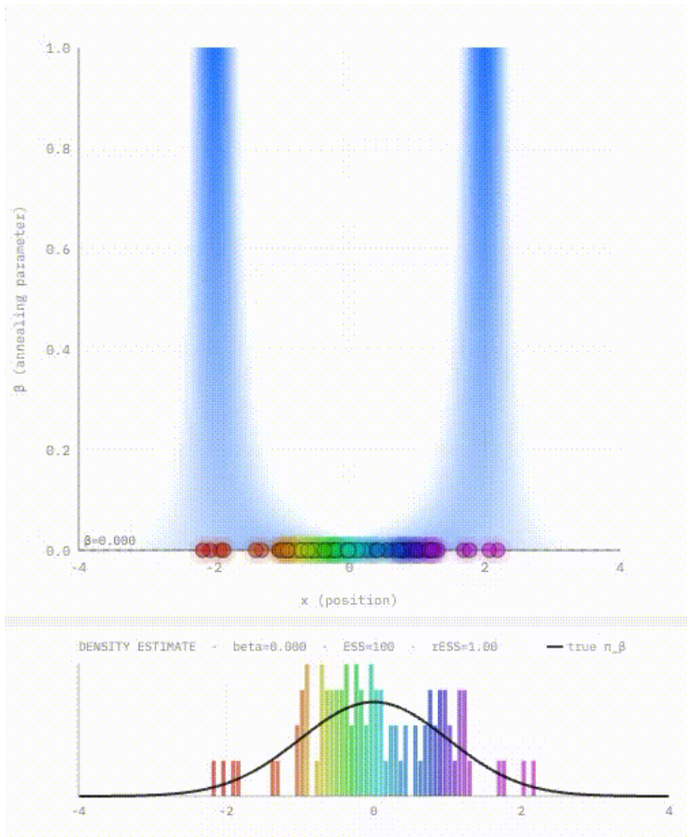
- We simultaneously evolve a system of particles to approximation the annealing path

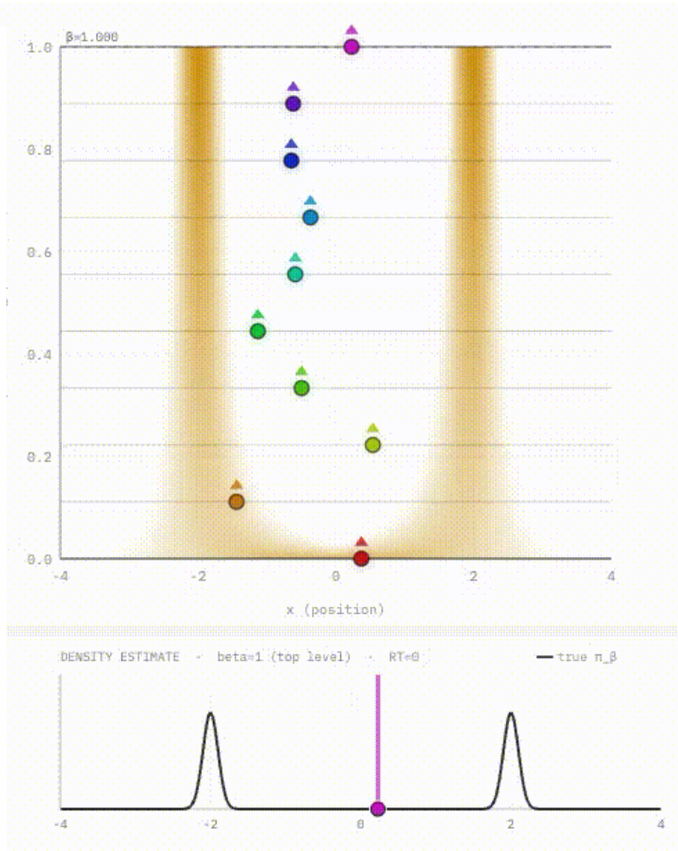
Sequential Annealing

- Construct path sequentially
- Samples generated in parallel

Parallel Annealing

- Construct path in parallel





SEQUENTIAL VERSUS PARALLEL ANNEALING

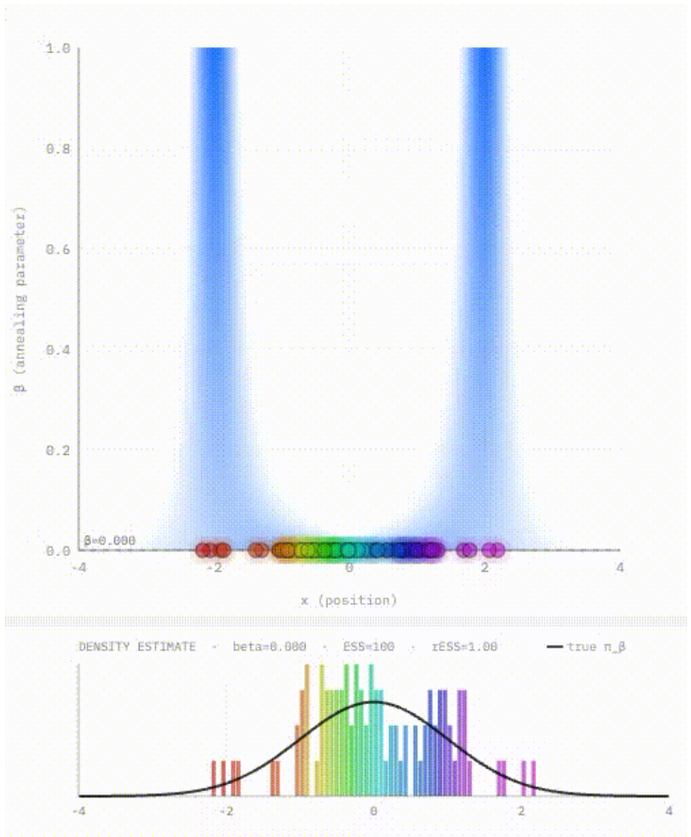
- We simultaneously evolve a system of particles to approximation the annealing path

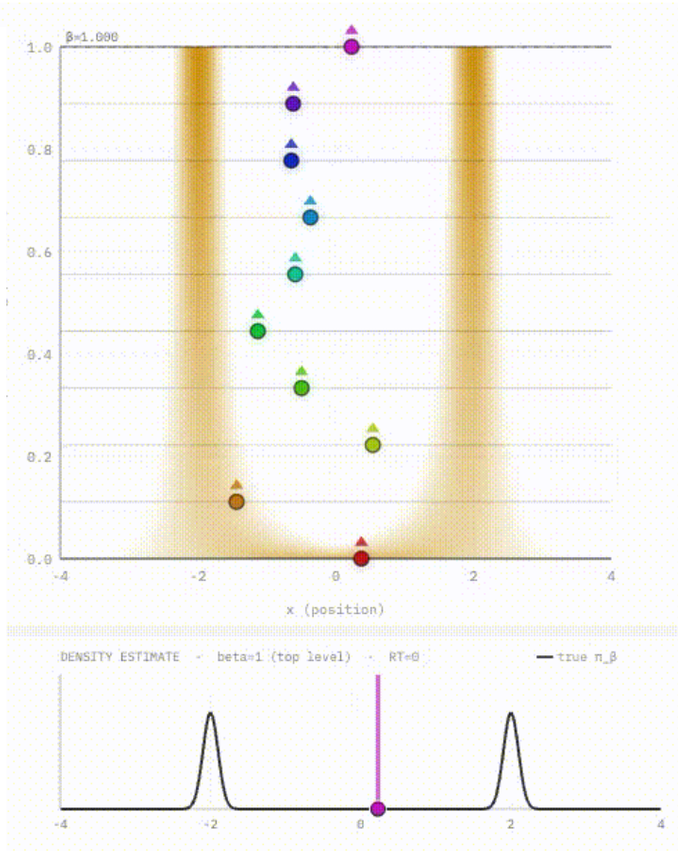
Sequential Annealing

- Construct path sequentially
- Samples generated in parallel

Parallel Annealing

- Construct path in parallel
- Samples generated in sequentially





SEQUENTIAL VERSUS PARALLEL ANNEALING

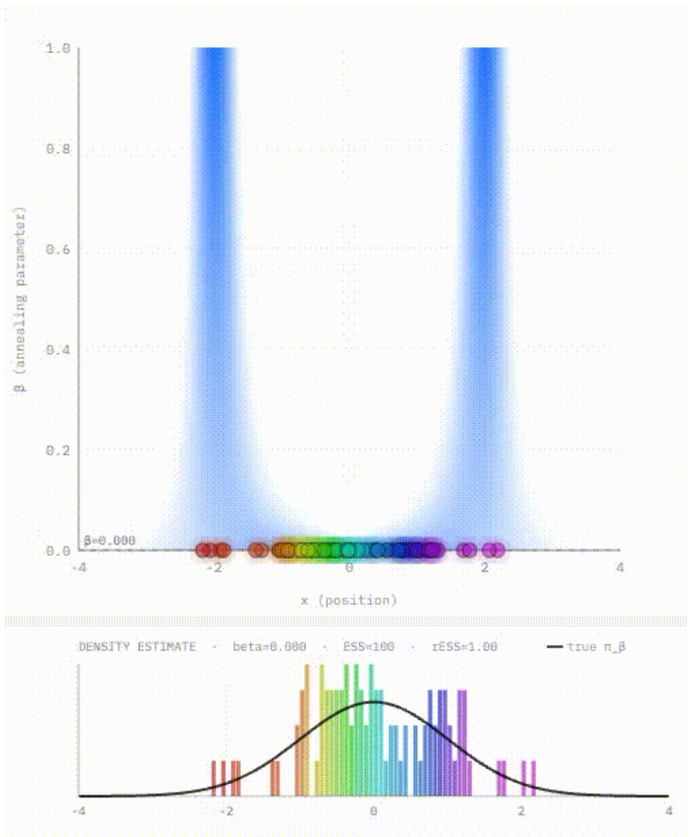
- We simultaneously evolve a system of particles to approximation the annealing path

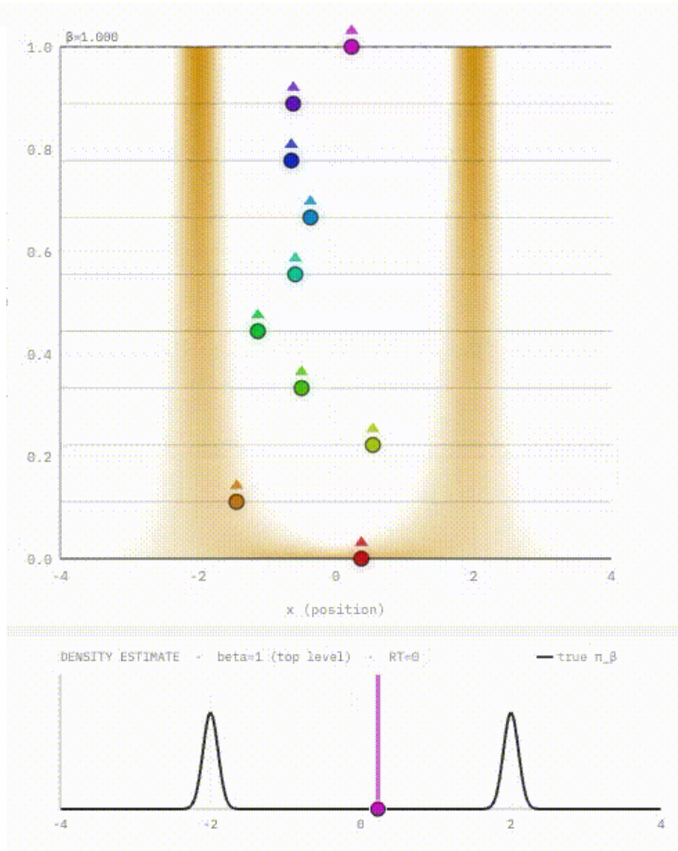
Sequential Annealing

- Construct path sequentially
- Samples generated in parallel
- Built using importance sampling

Parallel Annealing

- Construct path in parallel
- Samples generated in sequentially





SEQUENTIAL VERSUS PARALLEL ANNEALING

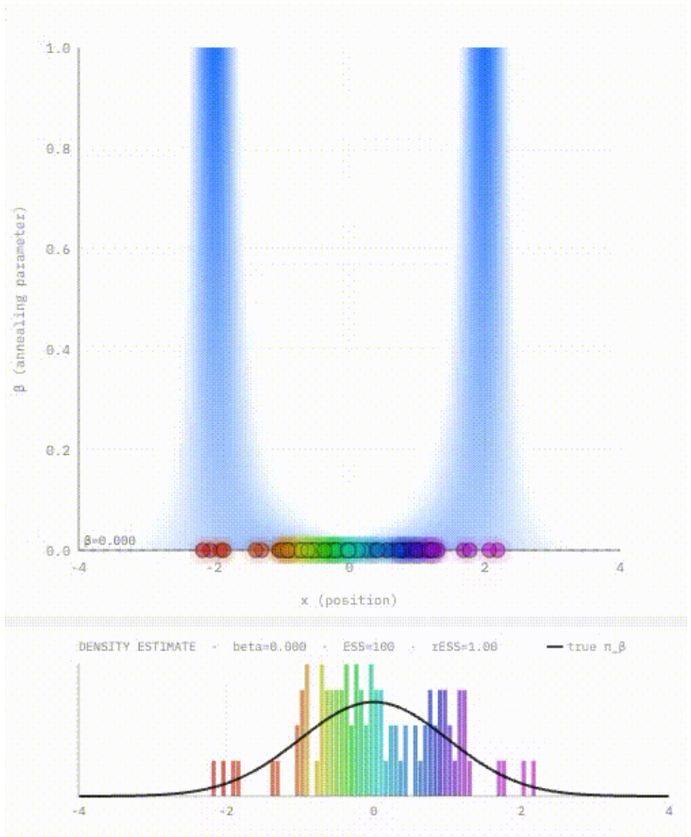
- We simultaneously evolve a system of particles to approximation the annealing path

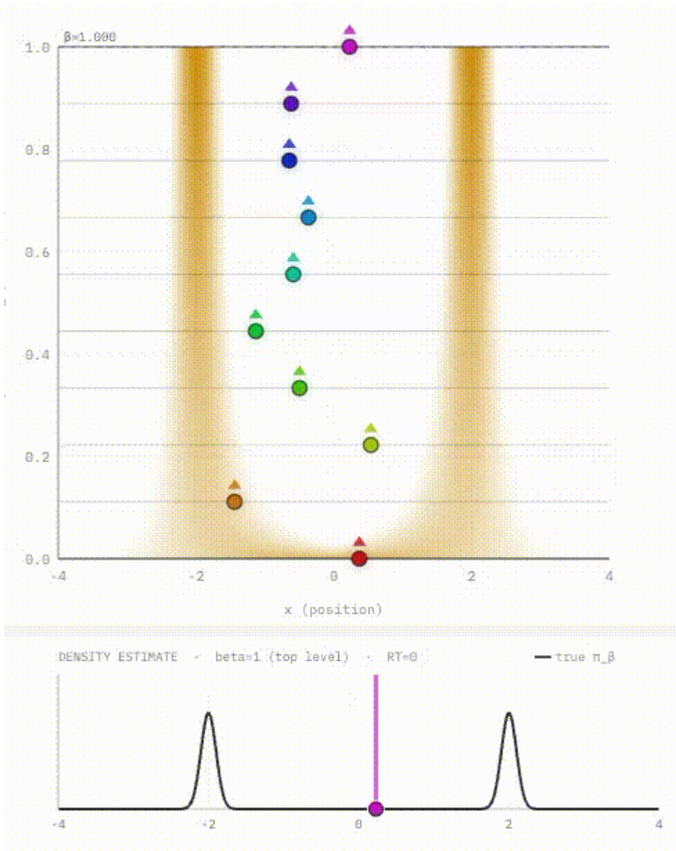
Sequential Annealing

- Construct path sequentially
- Samples generated in parallel
- Built using importance sampling

Parallel Annealing

- Construct path in parallel
- Samples generated in sequentially
- Built using MCMC





SEQUENTIAL VERSUS PARALLEL ANNEALING

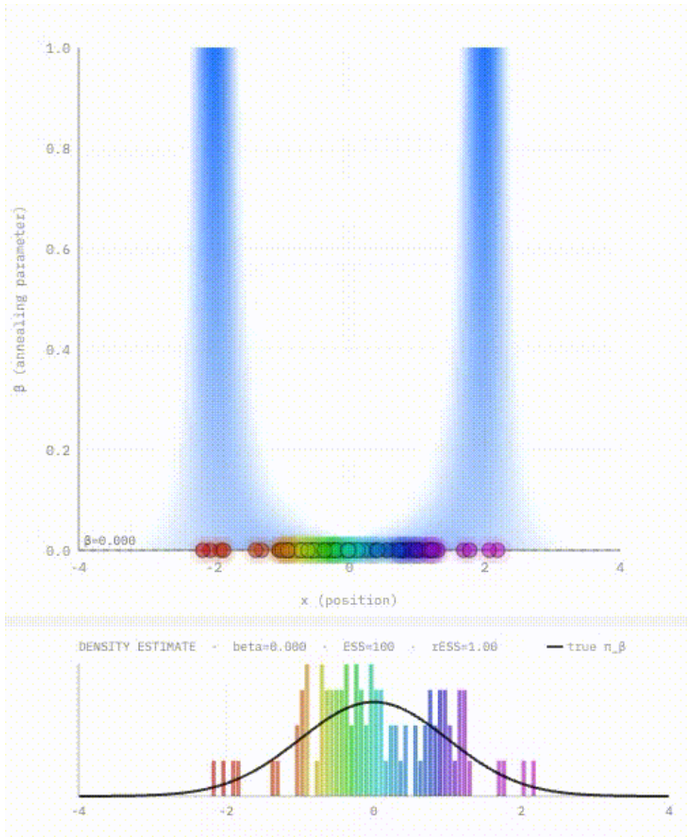
- We simultaneously evolve a system of particles to approximation the annealing path

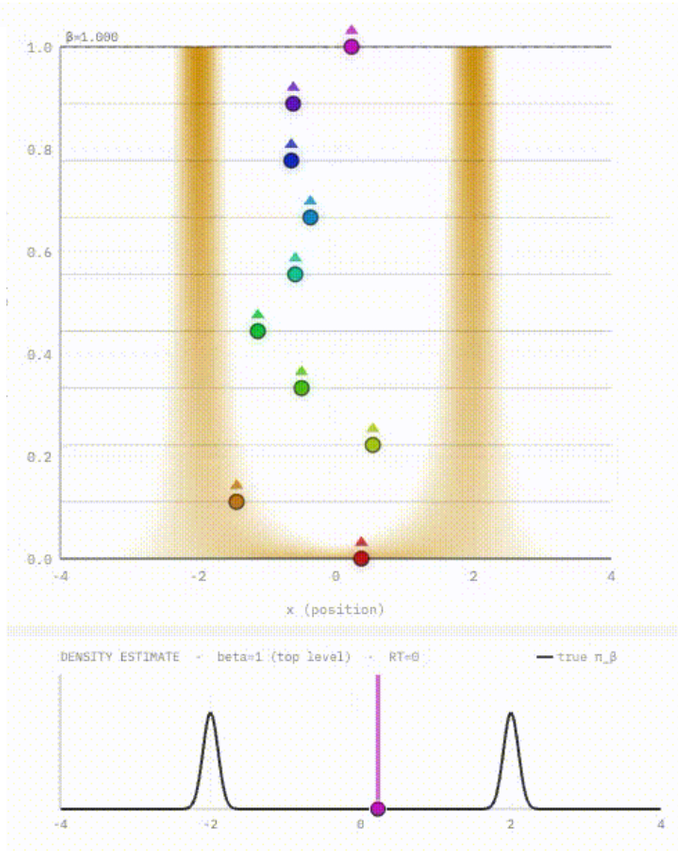
Sequential Annealing

- Construct path sequentially
- Samples generated in parallel
- Built using importance sampling
- Interacting → Sequential Monte Carlo Samplers

Parallel Annealing

- Construct path in parallel
- Samples generated in sequentially
- Built using MCMC





SEQUENTIAL VERSUS PARALLEL ANNEALING

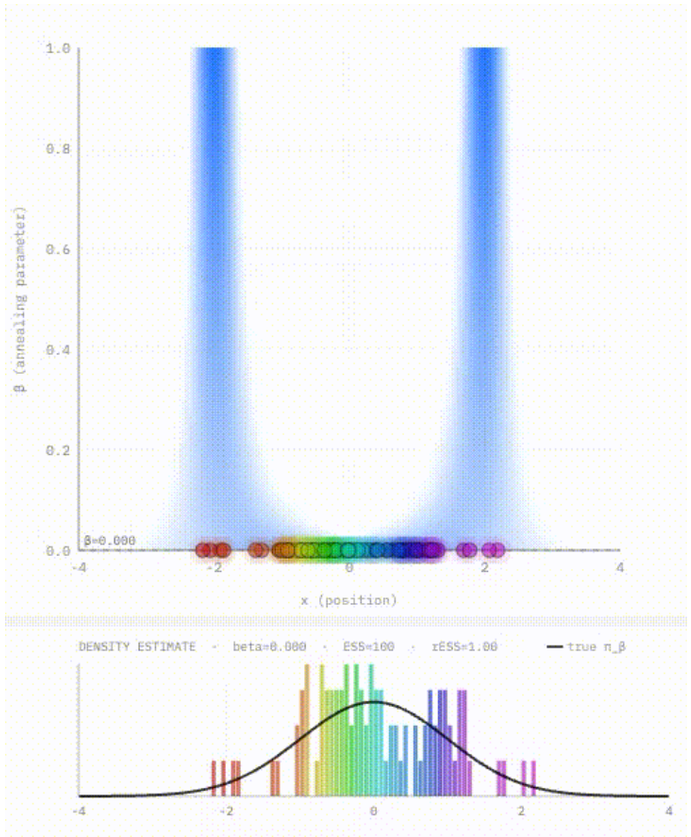
- We simultaneously evolve a system of particles to approximation the annealing path

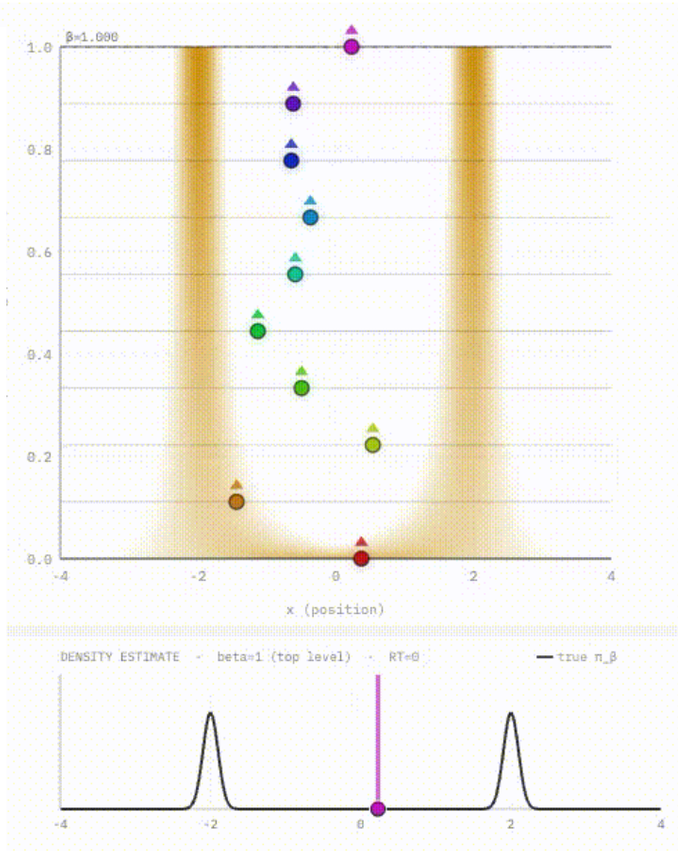
Sequential Annealing

- Construct path sequentially
- Samples generated in parallel
- Built using importance sampling
- Interacting → Sequential Monte Carlo Samplers

Parallel Annealing

- Construct path in parallel
- Samples generated in sequentially
- Built using MCMC
- Interacting → Parallel Tempering





SEQUENTIAL VERSUS PARALLEL ANNEALING

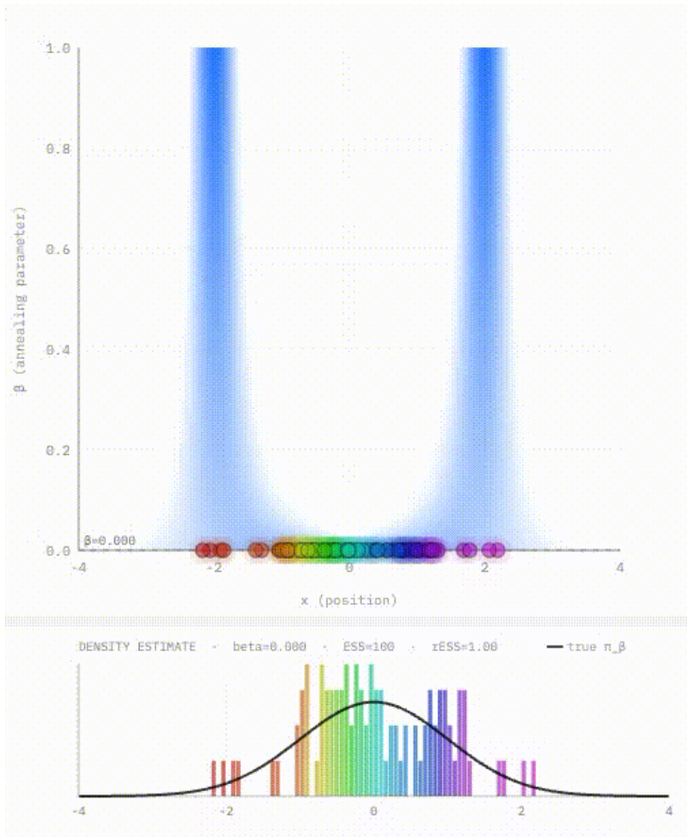
- We simultaneously evolve a system of particles to approximation the annealing path

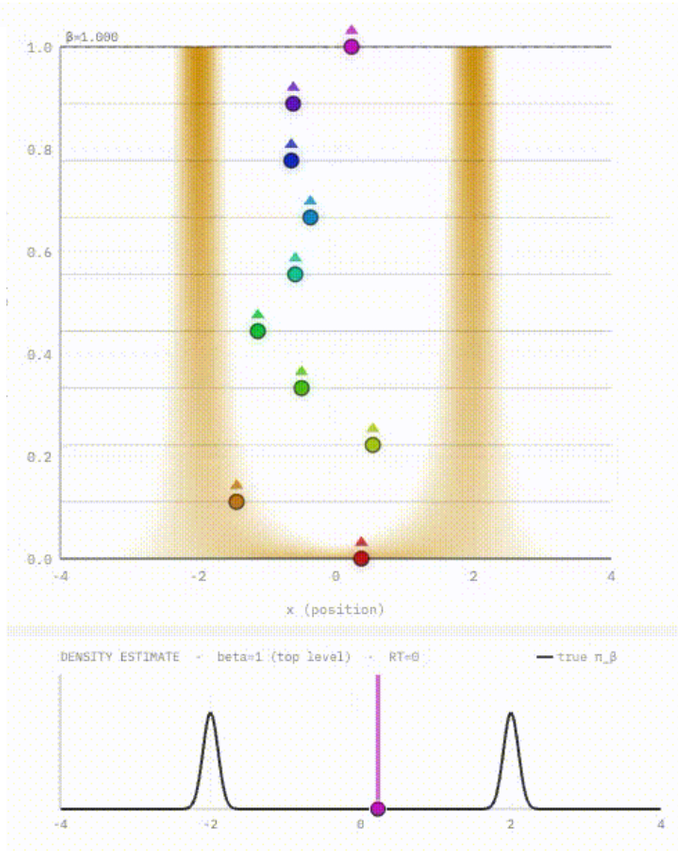
Sequential Annealing

- Construct path sequentially
- Samples generated in parallel
- Built using importance sampling
- Interacting → Sequential Monte Carlo Samplers
- Distributed → Simulated sampling

Parallel Annealing

- Construct path in parallel
- Samples generated in sequentially
- Built using MCMC
- Interacting → Parallel Tempering





SEQUENTIAL VERSUS PARALLEL ANNEALING

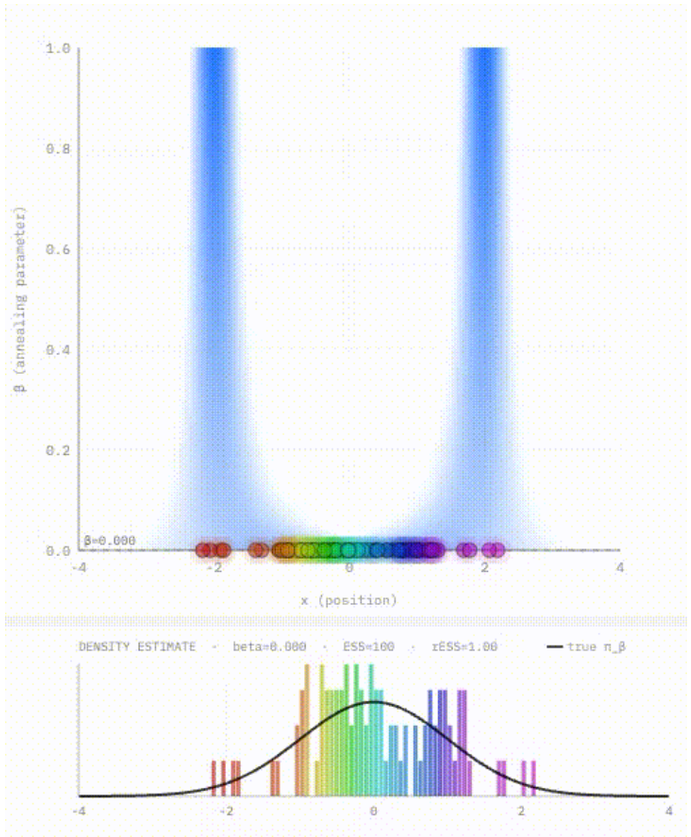
- We simultaneously evolve a system of particles to approximation the annealing path

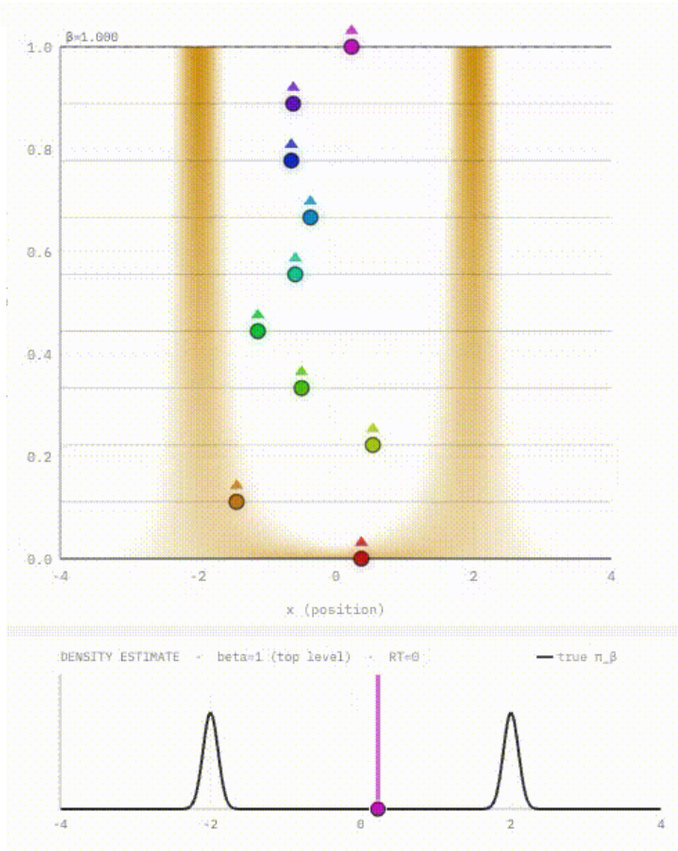
Sequential Annealing

- Construct path sequentially
- Samples generated in parallel
- Built using importance sampling
- Interacting → Sequential Monte Carlo Samplers
- Distributed → Simulated sampling

Parallel Annealing

- Construct path in parallel
- Samples generated in sequentially
- Built using MCMC
- Interacting → Parallel Tempering
- Distributed → Annealed Importance Sampling





PLAN FOR TERM

PLAN FOR TERM

- For the next 5 lectures we will do the following:

PLAN FOR TERM

- For the next 5 lectures we will do the following:
  - **Parallel Annealing:**

PLAN FOR TERM

- For the next 5 lectures we will do the following:
  - **Parallel Annealing:**
    - Parallel Tempering (PT)

PLAN FOR TERM

- For the next 5 lectures we will do the following:
  - **Parallel Annealing:**
    - Parallel Tempering (PT)
    - Simualted Tempering (ST)

PLAN FOR TERM

- For the next 5 lectures we will do the following:
  - **Parallel Annealing:**
    - Parallel Tempering (PT)
    - Simualted Tempering (ST)
  - **Sequential annealing:**

PLAN FOR TERM

- For the next 5 lectures we will do the following:
  - **Parallel Annealing:**
    - Parallel Tempering (PT)
    - Simualted Tempering (ST)
  - **Sequential annealing:**
    - Annealed importance sampling (AIS)

PLAN FOR TERM

- For the next 5 lectures we will do the following:
  - **Parallel Annealing:**
    - Parallel Tempering (PT)
    - Simualted Tempering (ST)
  - **Sequential annealing:**
    - Annealed importance sampling (AIS)
    - (Annealed) Sequential Monte Carlo samplers (ASMC)

PLAN FOR TERM

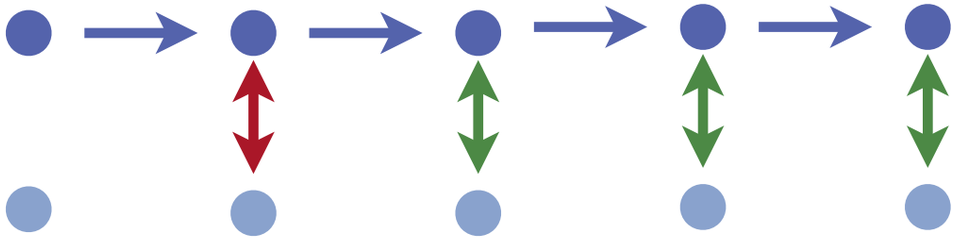
- For the next 5 lectures we will do the following:
  - **Parallel Annealing:**
    - Parallel Tempering (PT)
    - Simualted Tempering (ST)
  - **Sequential annealing:**
    - Annealed importance sampling (AIS)
    - (Annealed) Sequential Monte Carlo samplers (ASMC)
  - **Tuning annealing algorithms**

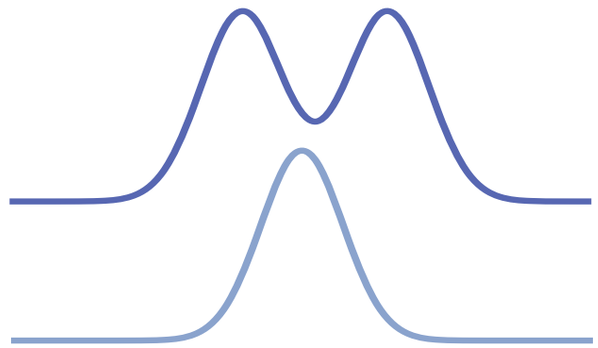
PLAN FOR TERM

- For the next 5 lectures we will do the following:
  - **Parallel Annealing:**
    - Parallel Tempering (PT)
    - Simualted Tempering (ST)
  - **Sequential annealing:**
    - Annealed importance sampling (AIS)
    - (Annealed) Sequential Monte Carlo samplers (ASMC)
  - **Tuning annealing algorithms**
  - **Normalising constant estimation**

RECALL STABILISED MCMC

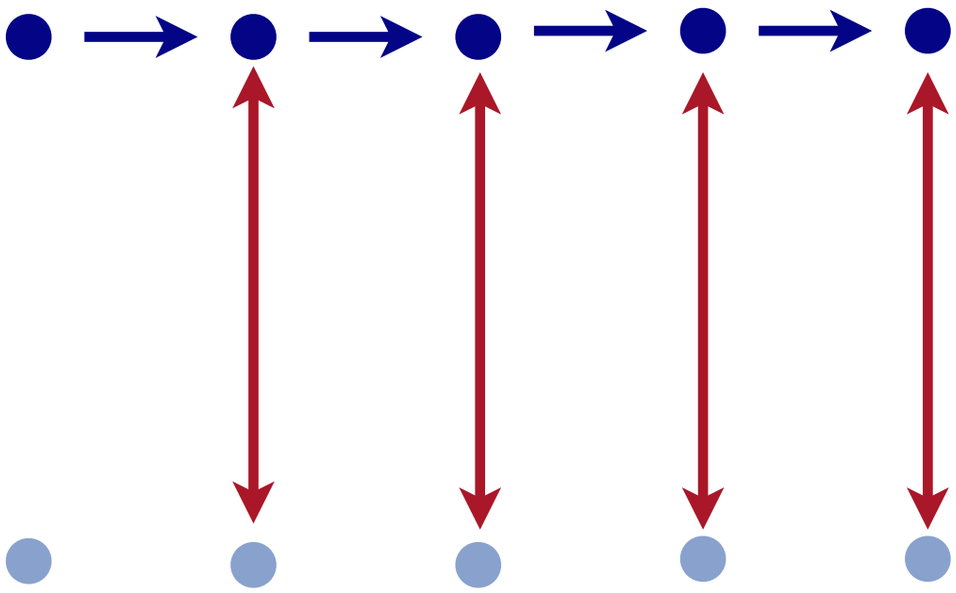
- Referenced stabilised MCMC is **stable** when reference is close to target
  - Acceptance rate is stable

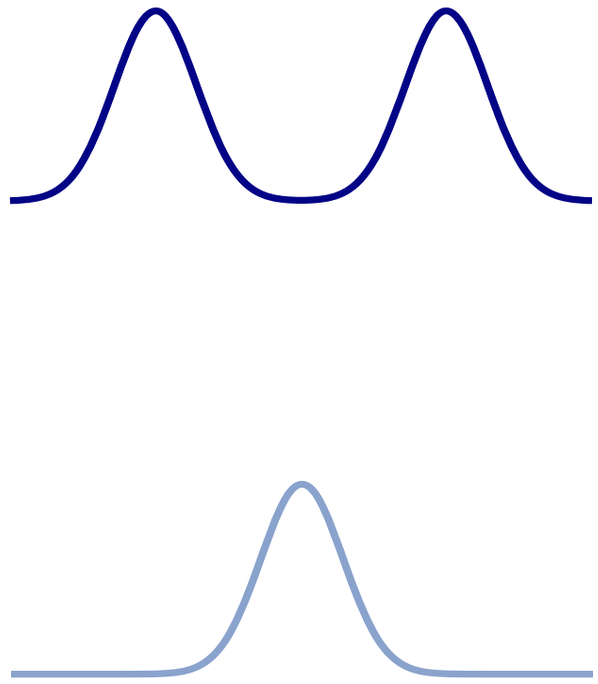




RECALL STABILISED MCMC

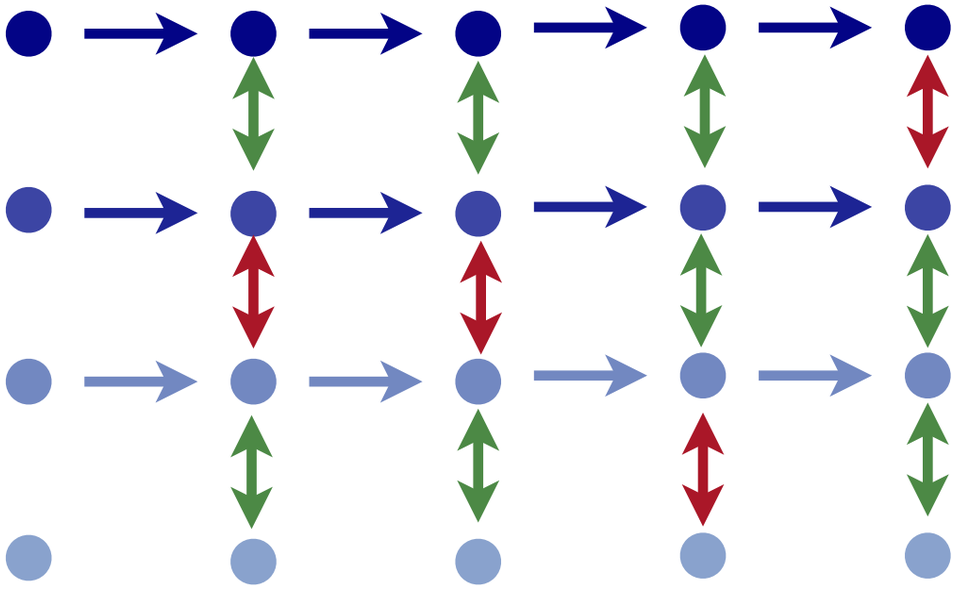
- Referenced stabilised MCMC is **stable** when reference is close to target
  - Acceptance rate is stable
- Referenced stabilised MCMC is **fragile** when the reference share little support with target
  - Most proposed refernce samples are rejected

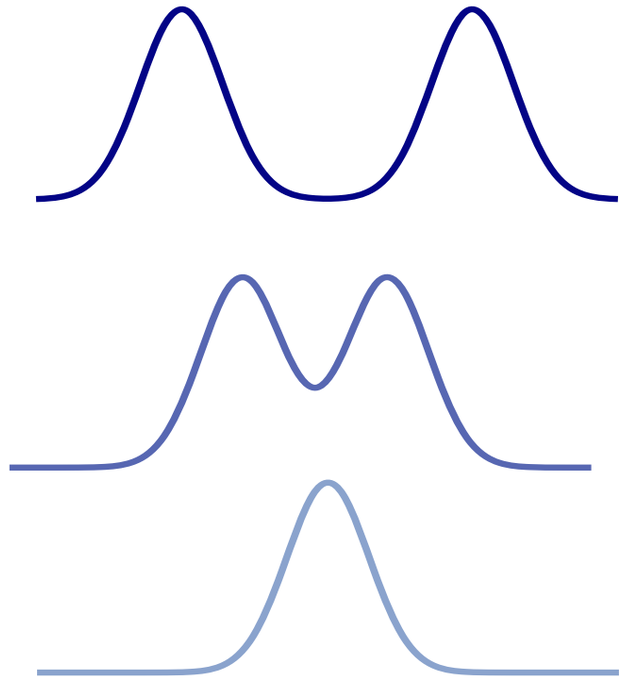




RECALL STABILISED MCMC

- Referenced stabilised MCMC is **stable** when reference is close to target
  - Acceptance rate is stable
- Referenced stabilised MCMC is **fragile** when the reference share little support with target
  - Most proposed refernce samples are rejected
- Introduce multiple interpolating distributions with enough overlap to maintain stability





PARALLEL TEMPERING

PARALLEL TEMPERING

- Uses stabilised MCMC to target annealing distributions in **parallel**

PARALLEL TEMPERING

- Uses stabilised MCMC to target annealing distributions in **parallel**

Target X_t^N

$\vdots$

X_t^1

Reference X_t^0

PARALLEL TEMPERING

- Uses stabilised MCMC to target annealing distributions in **parallel**

Target X_t^N ●

⋮ ●

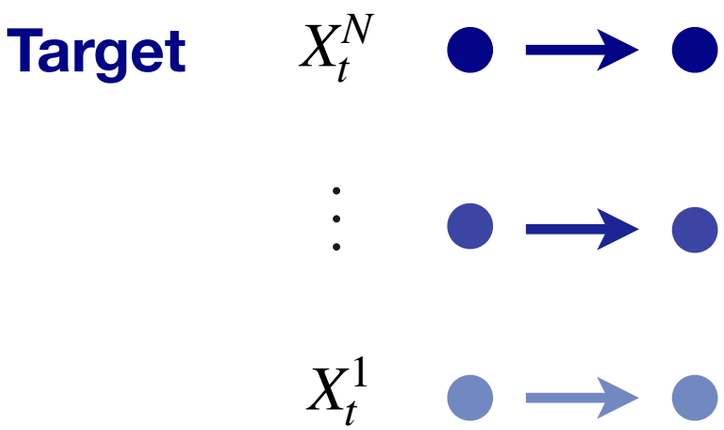
X_t^1 ●

Reference X_t^0 ●

- **Algorithm:** given N states at iteration $t-1$

PARALLEL TEMPERING

- Uses stabilised MCMC to target annealing distributions in **parallel**

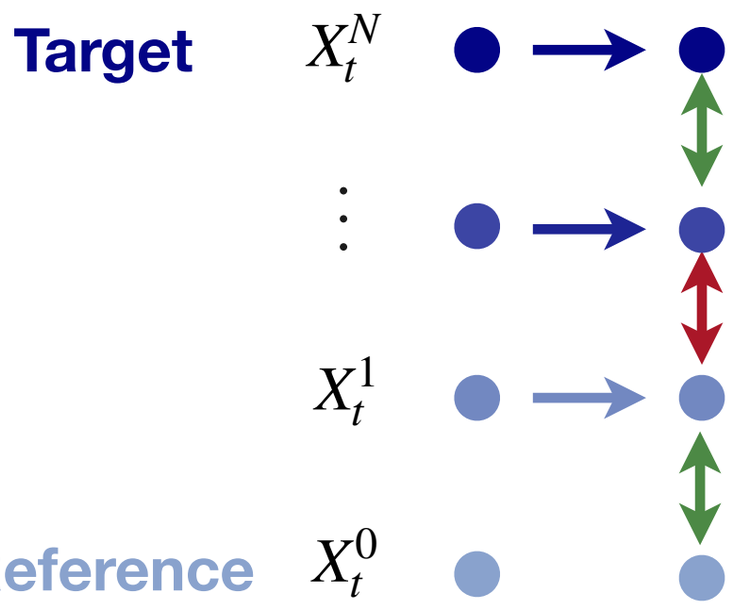


- **Algorithm:** given N states at iteration $t-1$

    1. **Local exploration:** Update n-th component with MCMC move for π_{β_n}

PARALLEL TEMPERING

- Uses stabilised MCMC to target annealing distributions in **parallel**

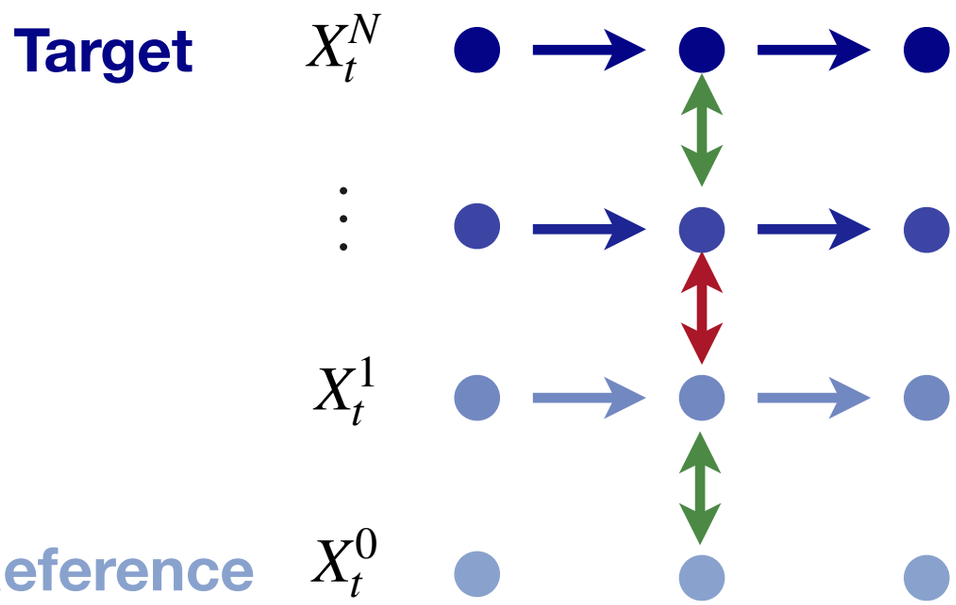


- **Algorithm:** given N states at iteration $t-1$

  1. **Local exploration:** Update n-th component with MCMC move for π_{β_n}
  2. **Communication:** Propose swaps for adjacent components
     - **Accept** / **Reject** according to Metropolis-Hastings criterion

PARALLEL TEMPERING

- Uses stabilised MCMC to target annealing distributions in **parallel**

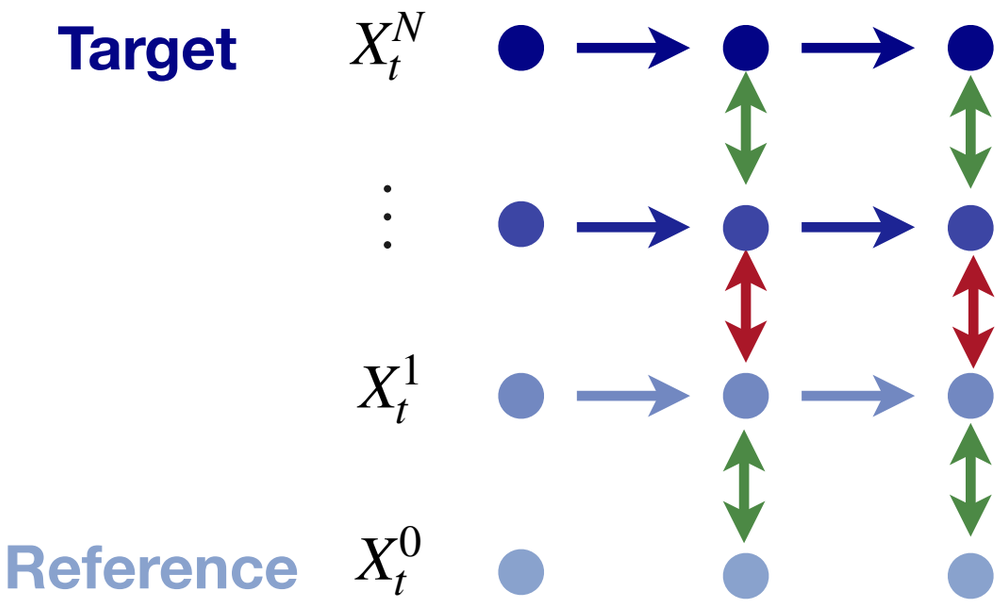


- **Algorithm:** given N states at iteration $t-1$

  1. **Local exploration:** Update n-th component with MCMC move for π_{β_n}
  2. **Communication:** Propose swaps for adjacent components
     - **Accept** / **Reject** according to Metropolis-Hastings criterion

PARALLEL TEMPERING

- Uses stabilised MCMC to target annealing distributions in **parallel**

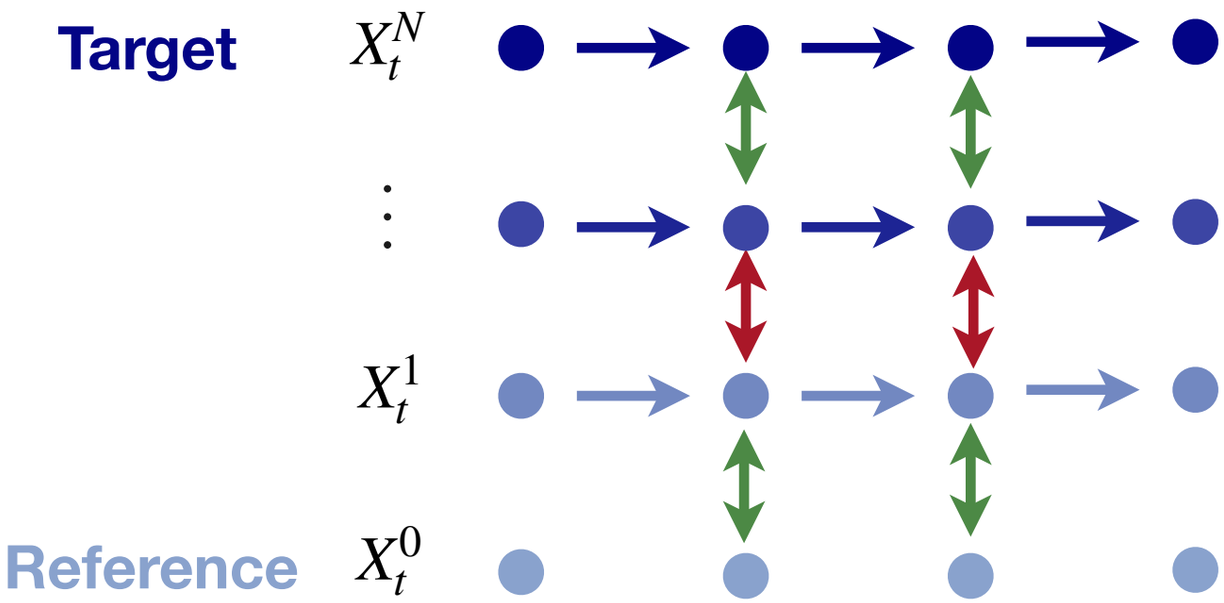


- **Algorithm:** given N states at iteration $t-1$

1. **Local exploration:** Update n-th component with MCMC move for π_{β_n}
2. **Communication:** Propose swaps for adjacent components
   - **Accept** / **Reject** according to Metropolis-Hastings criterion

PARALLEL TEMPERING

- Uses stabilised MCMC to target annealing distributions in **parallel**

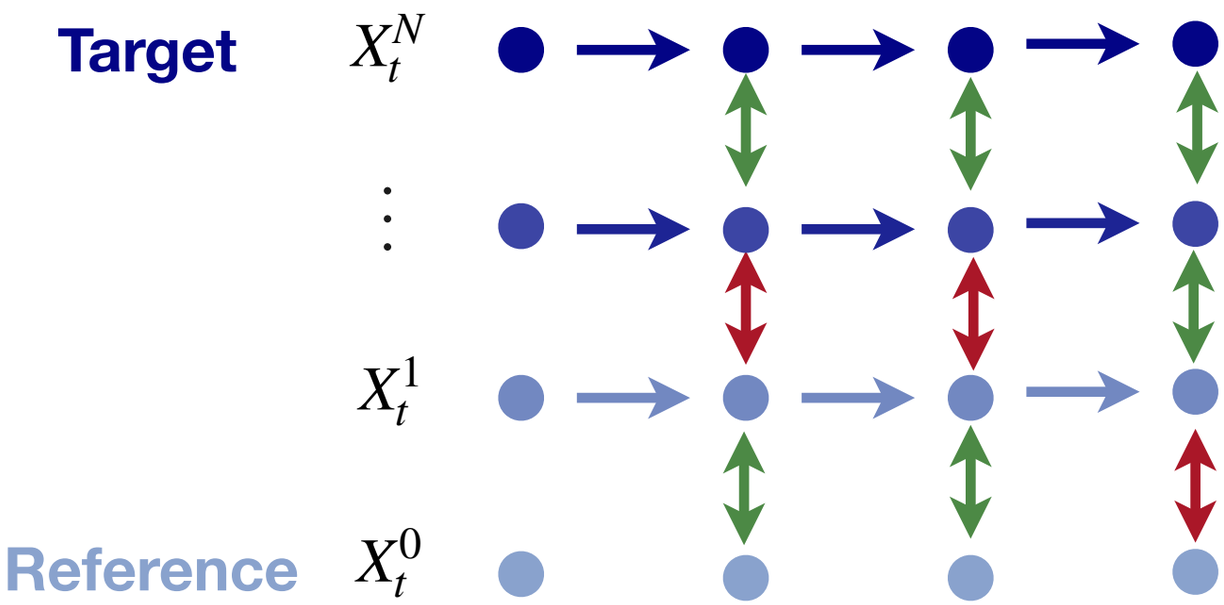


- **Algorithm:** given N states at iteration $t-1$

1. **Local exploration:** Update n-th component with MCMC move for π_{β_n}
2. **Communication:** Propose swaps for adjacent components
   - **Accept** / **Reject** according to Metropolis-Hastings criterion

PARALLEL TEMPERING

- Uses stabilised MCMC to target annealing distributions in **parallel**

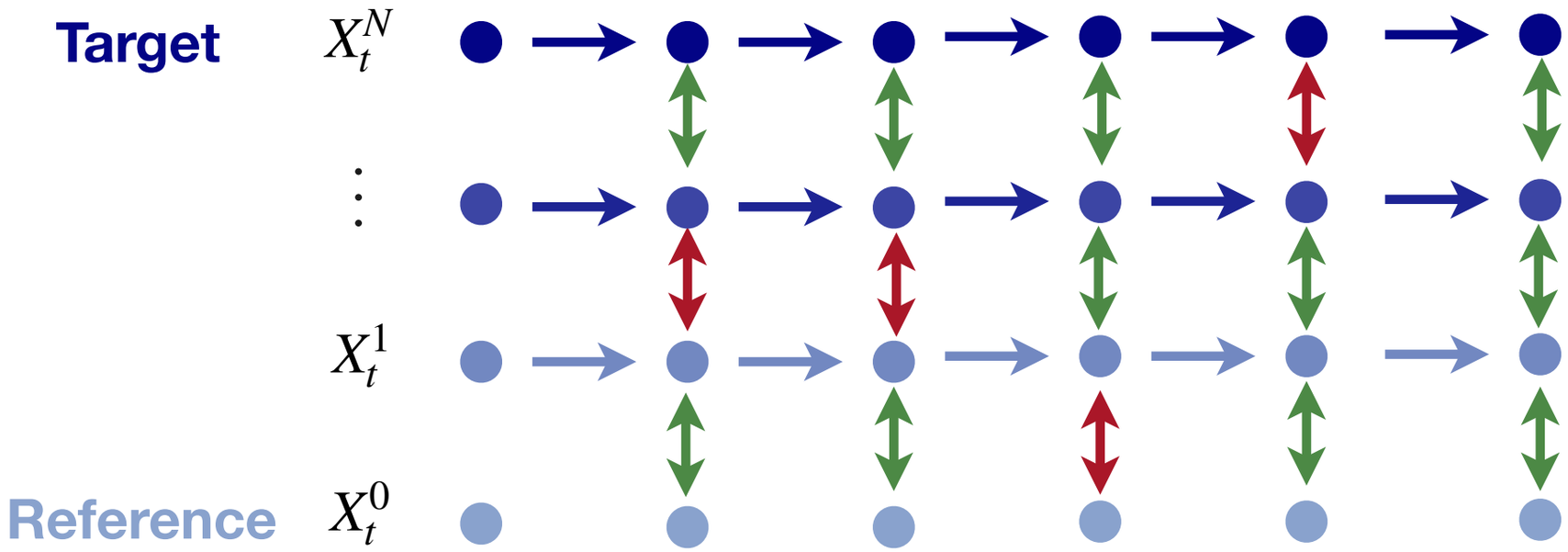


- **Algorithm:** given N states at iteration $t-1$

1. **Local exploration:** Update n-th component with MCMC move for π_{β_n}
2. **Communication:** Propose swaps for adjacent components
   - **Accept** / **Reject** according to Metropolis-Hastings criterion

PARALLEL TEMPERING

- Uses stabilised MCMC to target annealing distributions in **parallel**

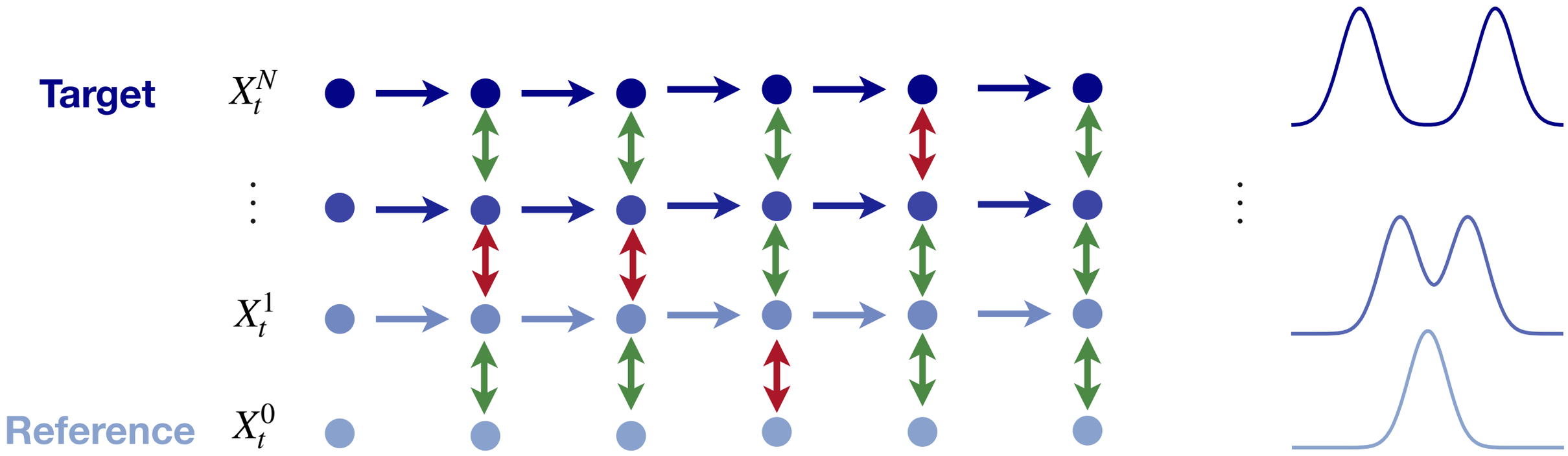


- **Algorithm:** given N states at iteration $t-1$

  1. **Local exploration:** Update n-th component with MCMC move for π_{β_n}
  2. **Communication:** Propose swaps for adjacent components
     - **Accept** / **Reject** according to Metropolis-Hastings criterion

PARALLEL TEMPERING

- Uses stabilised MCMC to target annealing distributions in **parallel**



- **Algorithm:** given N states at iteration $t-1$

1. **Local exploration:** Update n-th component with MCMC move for π_{β_n}
2. **Communication:** Propose swaps for adjacent components
   - **Accept** / **Reject** according to Metropolis-Hastings criterion

PARALLEL TEMPERING

PARALLEL TEMPERING

- Independently discovered in statistics and physics

PARALLEL TEMPERING

- Independently discovered in statistics and physics
  - Charles Geyer in 1991 in statistics

PARALLEL TEMPERING

- Independently discovered in statistics and physics
  - Charles Geyer in 1991 in statistics
  - Koji Hukushima and Koji Nemoto in 1995 in physics

PARALLEL TEMPERING

- Independently discovered in statistics and physics
  - Charles Geyer in 1991 in statistics
  - Koji Hukushima and Koji Nemoto in 1995 in physics

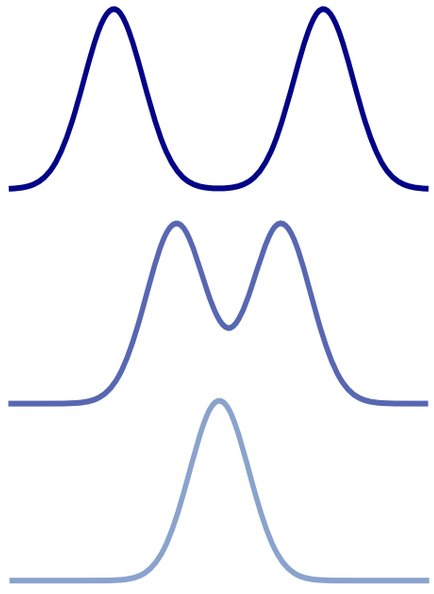
- Parallel tempering constructs MCMC chain $\mathbf{X}_t$ on $\mathbb{X}^{N+1}$ targetting $\pi_{\beta_0} \otimes \cdots \otimes \pi_{\beta_N}$

$$\mathbf{X}_t = \begin{pmatrix} X_t^N \\ \vdots \\ X_t^n \\ \vdots \\ X_t^0 \end{pmatrix} \begin{matrix} \sim \pi_{\beta_N} \\ \\ \sim \pi_{\beta_n} \\ \\ \sim \pi_{\beta_0} \end{matrix}$$

Target States

Interpolating States

Reference States

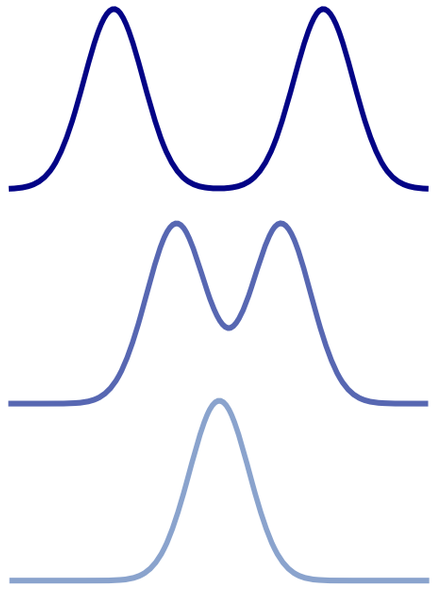


PARALLEL TEMPERING

- Independently discovered in statistics and physics
  - Charles Geyer in 1991 in statistics
  - Koji Hukushima and Koji Nemoto in 1995 in physics

- Parallel tempering constructs MCMC chain $\mathbf{X}_t$ on $\mathbb{X}^{N+1}$ targetting $\pi_{\beta_0} \otimes \cdots \otimes \pi_{\beta_N}$

$$\mathbf{X}_t = \begin{pmatrix} X_t^N \\ \vdots \\ X_t^n \\ \vdots \\ X_t^0 \end{pmatrix} \quad \begin{matrix} \sim \pi_{\beta_N} \\ \\ \sim \pi_{\beta_n} \\ \\ \sim \pi_{\beta_0} \end{matrix} \quad \begin{matrix} \textbf{Target States} \\ \\ \textbf{Interpolating States} \\ \\ \textbf{Reference States} \end{matrix}$$

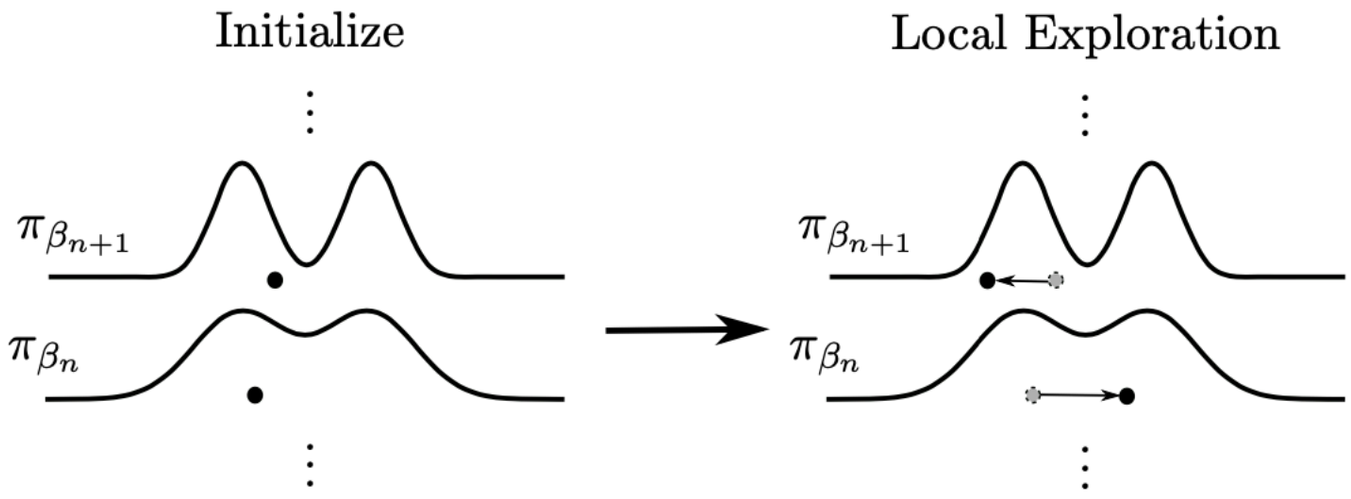


- Generate $\mathbf{X}_t$ from $\mathbf{X}_{t-1}$ using by applying the local exploration and communication moves

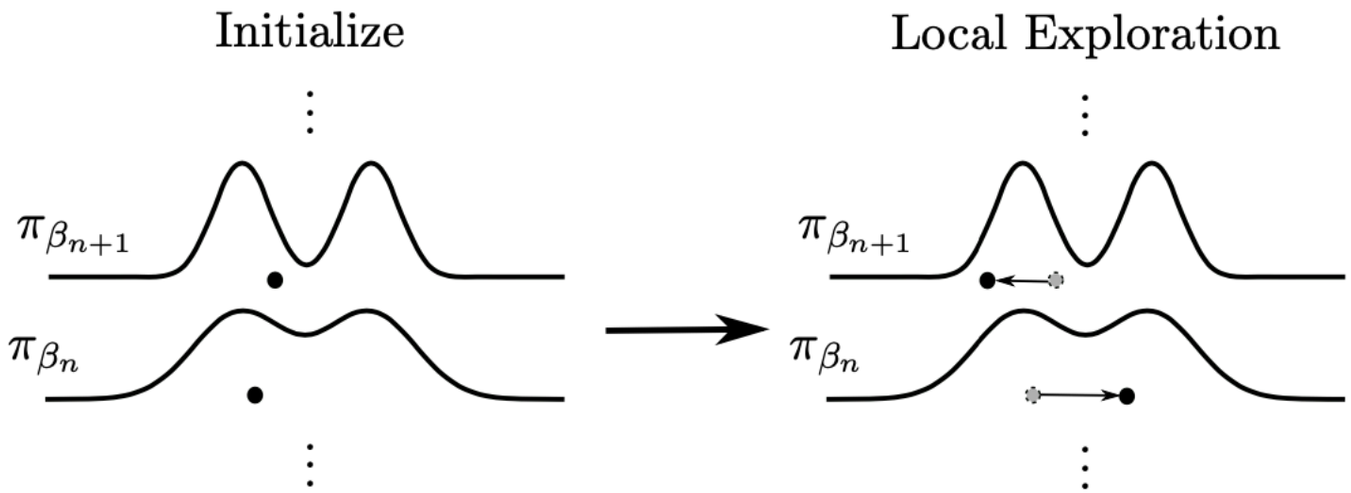
$$\mathbf{X}_t \sim \mathbf{K}_{\mathrm{PT}}(\mathbf{X}_{t-1}, \mathrm{d}\mathbf{x}') \qquad \mathbf{K}_{\mathrm{PT}} = \mathbf{K}_{\mathrm{expl}}\mathbf{K}_{\mathrm{comm}}$$

LOCAL EXPLORATION

LOCAL EXPLORATION

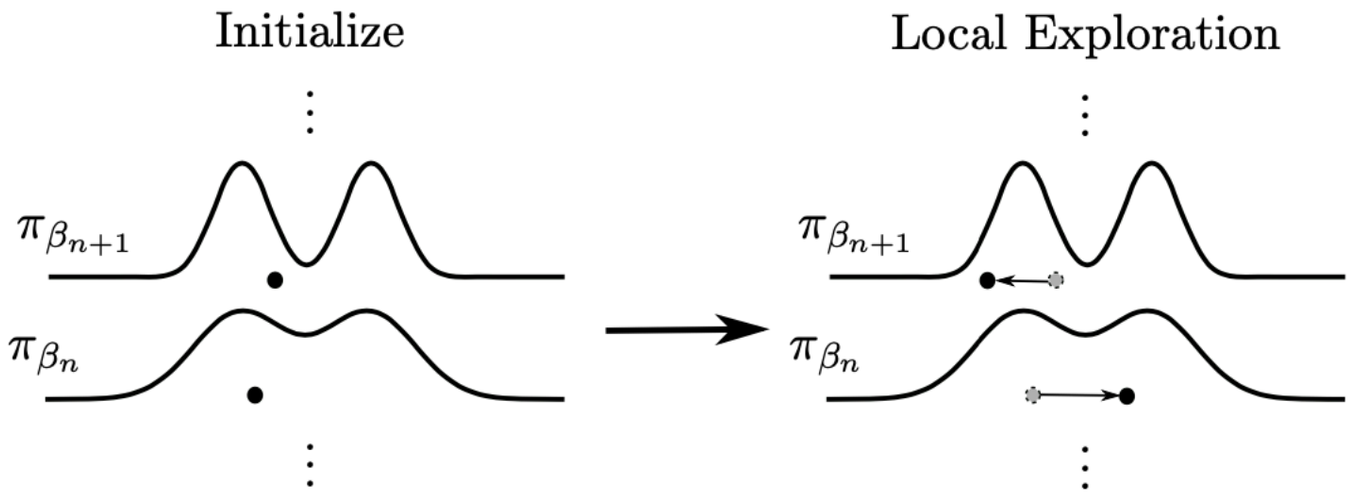


LOCAL EXPLORATION



- **Exploration:** Update components of $\mathbf{x} = (x^0, \ldots, x^N)$ in parallel using MCMC

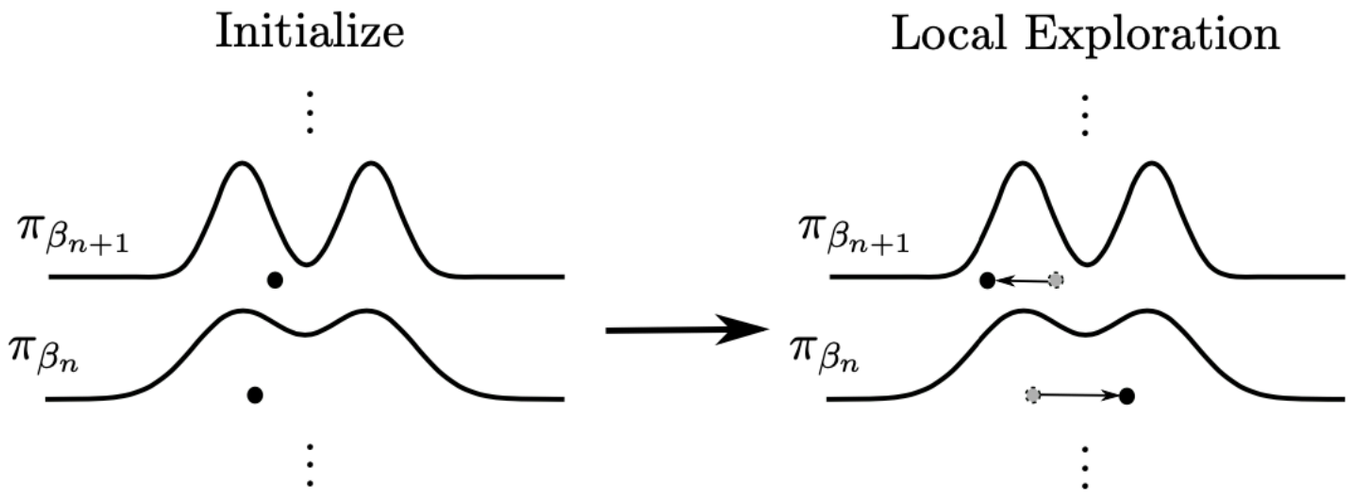
LOCAL EXPLORATION



- **Exploration:** Update components of $\mathbf{x} = (x^0, \ldots, x^N)$ in parallel using MCMC
- Update the x^n using the local move K_{β_n} targeting π_{β_n}

$$K_{\text{expl}}(\mathbf{x}, \mathrm{d}\mathbf{x}') = K_{\beta_0} \otimes \cdots \otimes K_{\beta_N}(\mathbf{x}, \mathrm{d}\mathbf{x}')$$

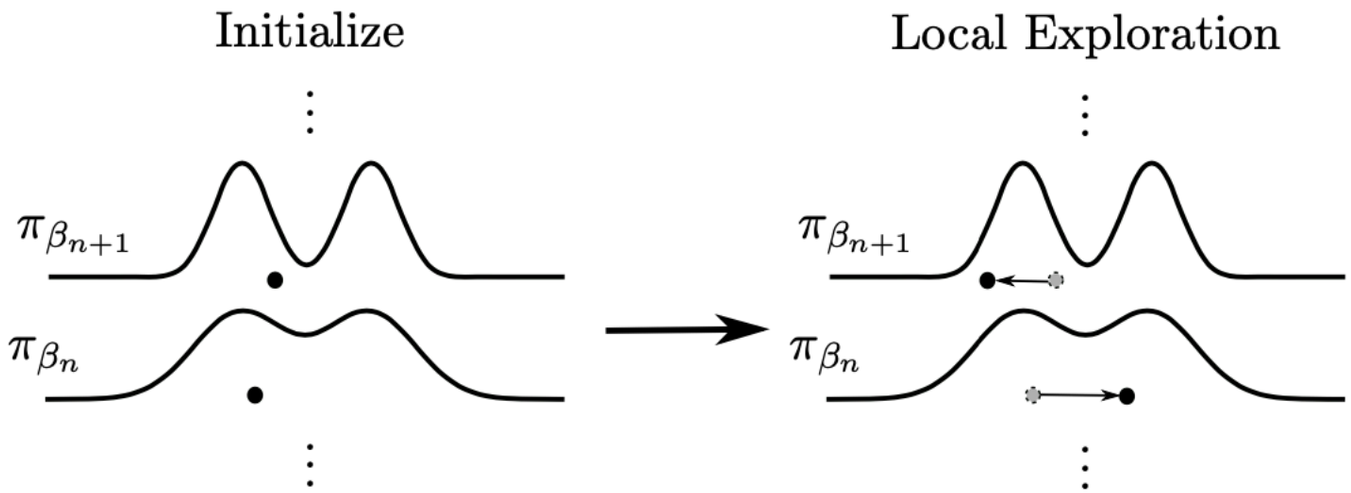
LOCAL EXPLORATION



- **Exploration:** Update components of $\mathbf{x} = (x^0, \ldots, x^N)$ in parallel using MCMC
- Update the x^n using the local move K_{β_n} targeting π_{β_n}
  - For example, Metropolis-Hastings, Gibbs sampling, RWM, MALA, HMC, etc.

$$K_{\text{expl}}(\mathbf{x}, \mathrm{d}\mathbf{x}') = K_{\beta_0} \otimes \cdots \otimes K_{\beta_N}(\mathbf{x}, \mathrm{d}\mathbf{x}')$$

LOCAL EXPLORATION



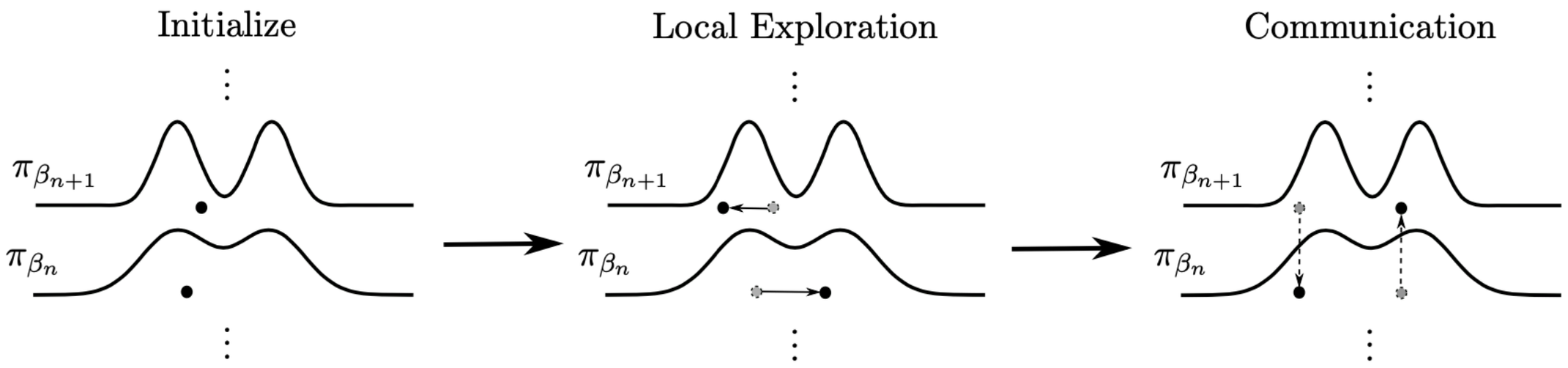
- **Exploration:** Update components of $\mathbf{x} = (x^0, \ldots, x^N)$ in parallel using MCMC

- Update the x^n using the local move K_{β_n} targeting π_{β_n}
  - For example, Metropolis-Hastings, Gibbs sampling, RWM, MALA, HMC, etc.
  - Assume that an independent sample from reference

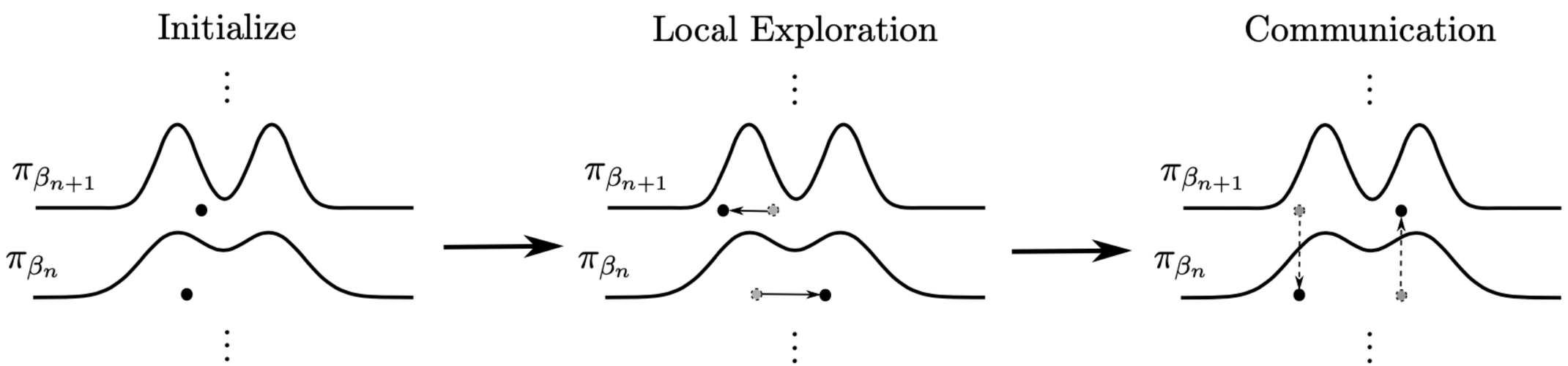
$$K_{\text{expl}}(\mathbf{x}, \mathrm{d}\mathbf{x}') = K_{\beta_0} \otimes \cdots \otimes K_{\beta_N}(\mathbf{x}, \mathrm{d}\mathbf{x}')$$

$$= \eta(\mathrm{d}x'^0) \prod_{n=1}^{N} K_{\beta_n}(x^n, \mathrm{d}x'^n)$$

SWAP MOVE


Initialize
Local Exploration
Communication
$\pi_{\beta_{n+1}}$
π_{β_n}
$\pi_{\beta_{n+1}}$
π_{β_n}
$\pi_{\beta_{n+1}}$
π_{β_n}

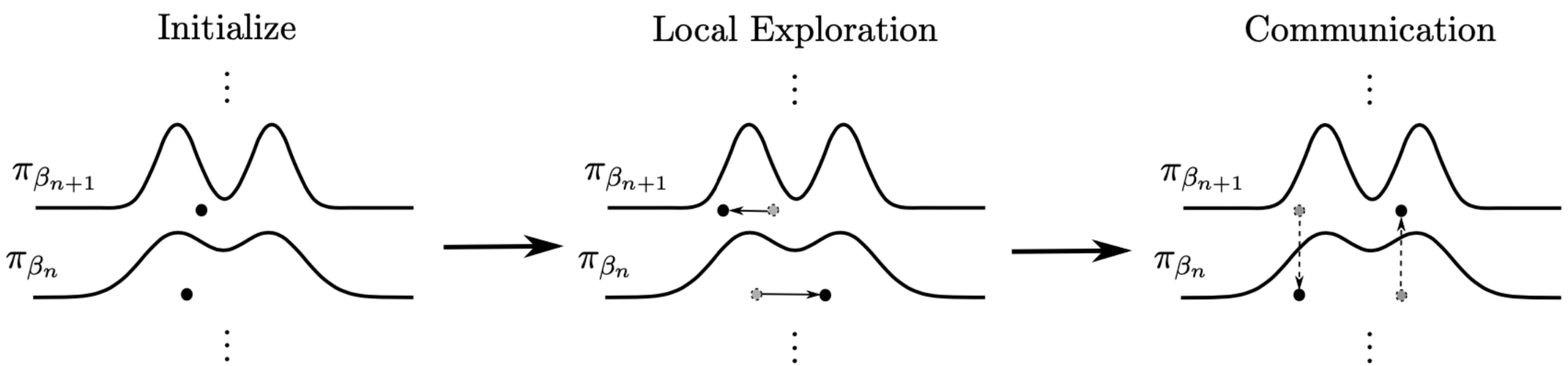
SWAP MOVE



- The n-th swap move takes the state $\mathbf{x}$ proposes $\mathbf{x}^{(n,n+1)}$ swapping components n and $n+1$

$$\mathbf{x}^{(n,n+1)} = (x^0, \ldots, x^{n-1}, x^{n+1}, x^n, x^{n+2}, \ldots, x^N)$$

SWAP MOVE



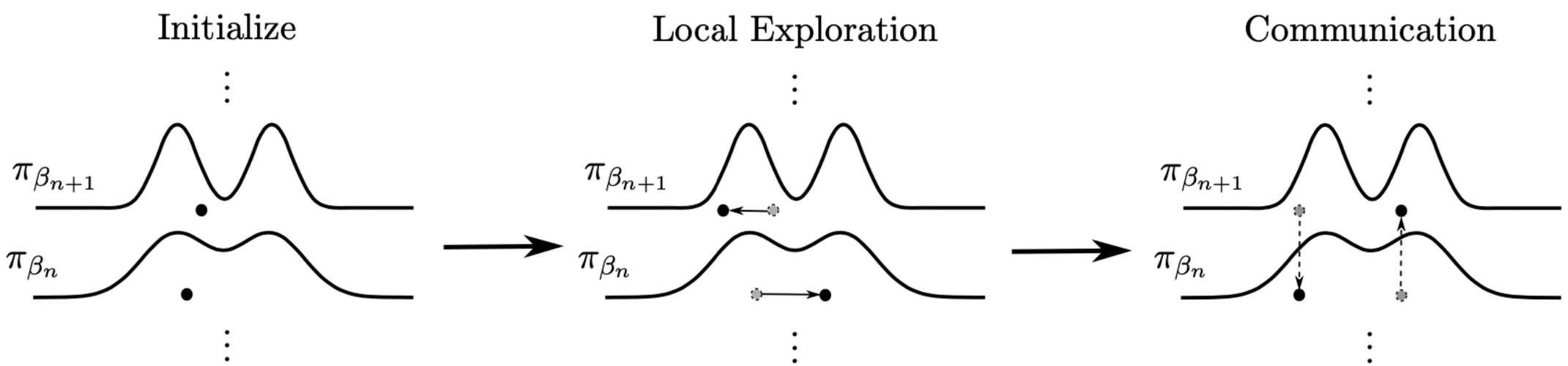
- The n-th swap move takes the state $\mathbf{x}$ proposes $\mathbf{x}^{(n,n+1)}$ swapping components n and $n+1$

$$\mathbf{x}^{(n,n+1)} = (x^0, \ldots, x^{n-1}, x^{n+1}, x^n, x^{n+2}, \ldots, x^N)$$

  - Accept/reject using Metropolis-Hastings:

$$\mathbf{K}^{(n,n+1)}(\mathbf{x}, \mathrm{d}\mathbf{x}') = \alpha^{(n,n+1)}(\mathbf{x})\delta_{\mathbf{x}^{(n,n+1)}}(\mathrm{d}\mathbf{x}') + (1 - \alpha^{(n,n+1)}(\mathbf{x})\delta_{\mathbf{x}}(\mathrm{d}\mathbf{x}')$$

SWAP MOVE



- The n-th swap move takes the state $\mathbf{x}$ proposes $\mathbf{x}^{(n,n+1)}$ swapping components n and $n+1$

$$\mathbf{x}^{(n,n+1)} = (x^0, \ldots, x^{n-1}, x^{n+1}, x^n, x^{n+2}, \ldots, x^N)$$

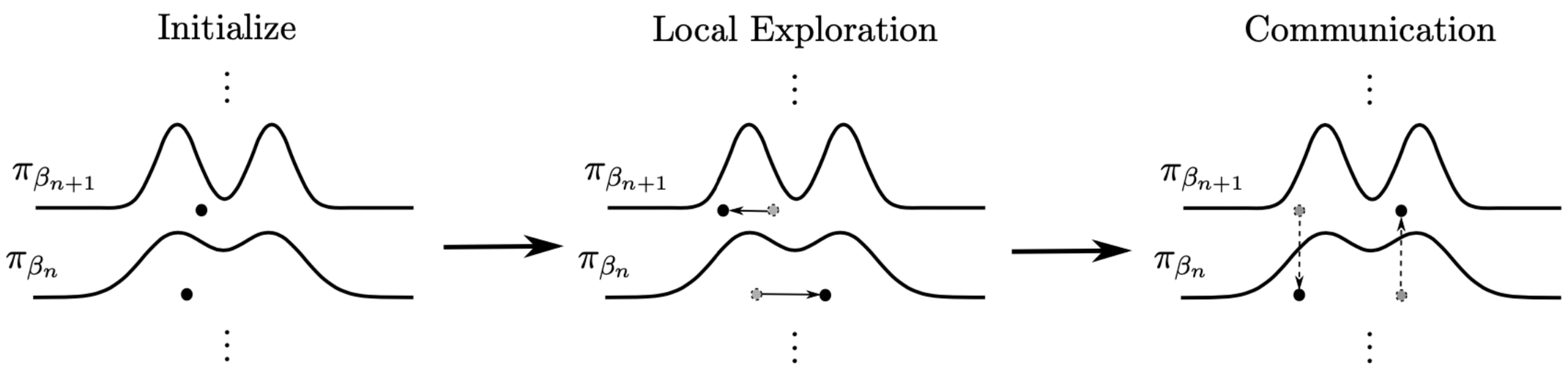
  - Accept/reject using Metropolis-Hastings:

$$\mathbf{K}^{(n,n+1)}(\mathbf{x}, \mathrm{d}\mathbf{x}') = \alpha^{(n,n+1)}(\mathbf{x})\delta_{\mathbf{x}^{(n,n+1)}}(\mathrm{d}\mathbf{x}') + (1 - \alpha^{(n,n+1)}(\mathbf{x})\delta_{\mathbf{x}}(\mathrm{d}\mathbf{x}')$$

  - Acceptance probability only depends on likelihood:

$$\alpha^{(n,n+1)}(\mathbf{x}) = 1 \wedge \frac{\boldsymbol{\pi}(\mathbf{x}^{(n,n+1)})}{\boldsymbol{\pi}(\mathbf{x})}$$

SWAP MOVE



- The n-th swap move takes the state $\mathbf{x}$ proposes $\mathbf{x}^{(n,n+1)}$ swapping components n and $n+1$

$$\mathbf{x}^{(n,n+1)} = (x^0, \ldots, x^{n-1}, x^{n+1}, x^n, x^{n+2}, \ldots, x^N)$$

  - Accept/reject using Metropolis-Hastings:

$$\mathbf{K}^{(n,n+1)}(\mathbf{x}, \mathrm{d}\mathbf{x}') = \alpha^{(n,n+1)}(\mathbf{x})\delta_{\mathbf{x}^{(n,n+1)}}(\mathrm{d}\mathbf{x}') + (1 - \alpha^{(n,n+1)}(\mathbf{x})\delta_{\mathbf{x}}(\mathrm{d}\mathbf{x}')$$

  - Acceptance probability only depends on likelihood:

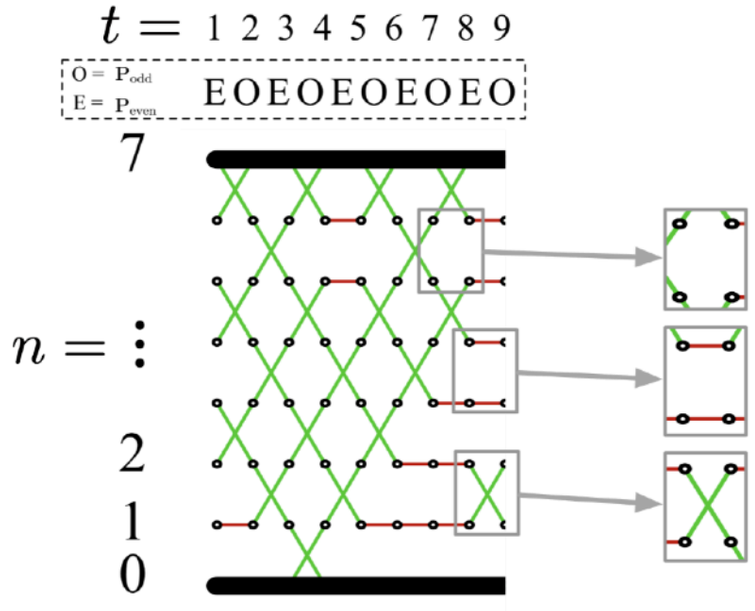
$$\alpha^{(n,n+1)}(\mathbf{x}) = 1 \wedge \frac{\boldsymbol{\pi}(\mathbf{x}^{(n,n+1)})}{\boldsymbol{\pi}(\mathbf{x})} = 1 \wedge \frac{\pi_{\beta_n}(x^{n+1})\pi_{\beta_{n+1}}(x^n)}{\pi_{\beta_n}(x^n)\pi_{\beta_{n+1}}(x^{n+1})}$$

COMMUNICATION

COMMUNICATION

- **Communication move**

$$\mathbf{K}_{\text{comm}} = \prod_{n=1}^{N} K^{(n,n+1)} P_t^{(n,n+1)}$$



Proposal indicators

$P_7^{(5,6)} = 0$

$P_8^{(3,4)} = 1$

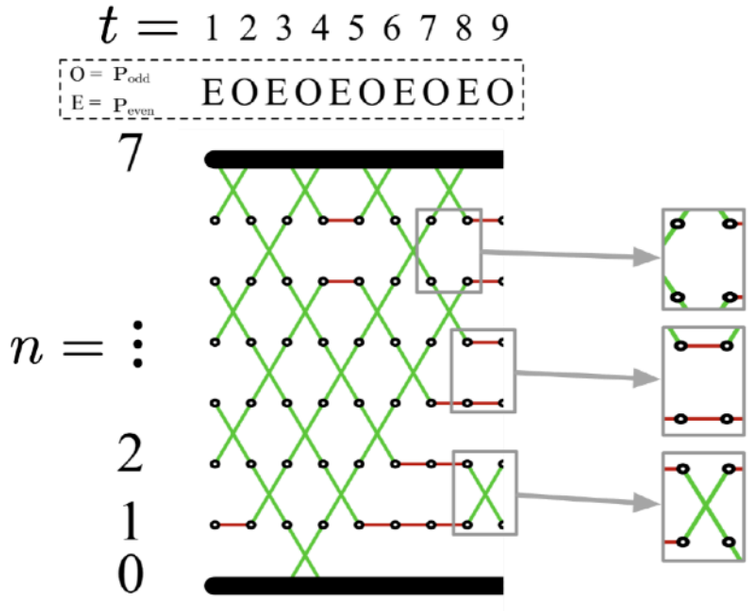
$P_8^{(1,2)} = 1$

COMMUNICATION

- **Communication move**

$$\mathbf{K}_{\text{comm}} = \prod_{n=1}^{N} K^{(n,n+1)} P_t^{(n,n+1)}$$

  - $P_t^{(n,n+1)} \in \{0,1\}$ indicated if the n-th swap is proposed

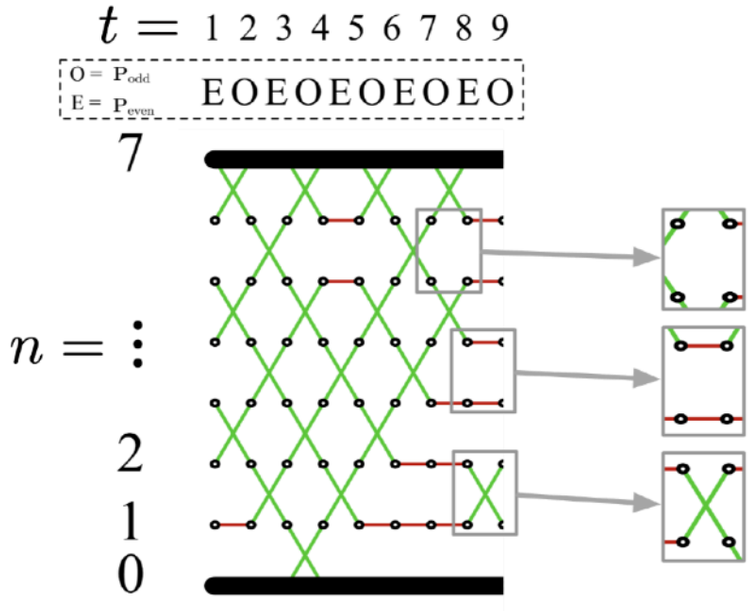


COMMUNICATION

- **Communication move**

$$\mathbf{K}_{\text{comm}} = \prod_{n=1}^{N} K^{(n,n+1)} P_t^{(n,n+1)}$$

  - $P_t^{(n,n+1)} \in \{0,1\}$ indicated if the n-th swap is proposed
  - Note that in general, the order matters



Proposal indicators

$P_7^{(5,6)} = 0$

$P_8^{(3,4)} = 1$

$P_8^{(1,2)} = 1$

COMMUNICATION

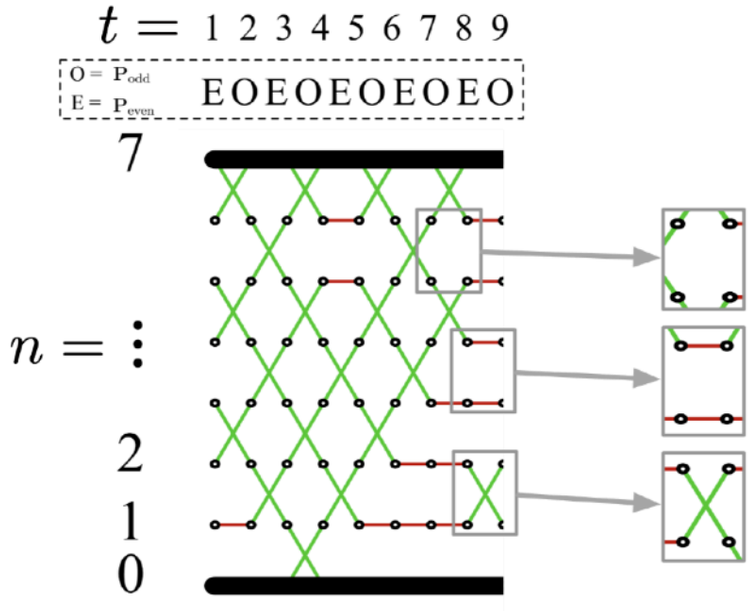
- **Communication move**

$$\mathbf{K}_{\text{comm}} = \prod_{n=1}^{N} K^{(n,n+1)} P_t^{(n,n+1)}$$

  - $P_t^{(n,n+1)} \in \{0,1\}$ indicated if the n-th swap is proposed
  - Note that in general, the order matters

- A special is odd and even swaps

$$\mathbf{K}_{\text{comm}} \in \{\mathbf{K}_{\text{odd}}, \mathbf{K}_{\text{even}}\}, \qquad \mathbf{K}_{\text{odd}} = \prod_{n \text{ odd}} \mathbf{K}^{(n,n+1)}, \quad \mathbf{K}_{\text{even}} = \prod_{n \text{ even}} \mathbf{K}^{(n,n+1)}$$



Proposal indicators

$P_7^{(5,6)} = 0$

$P_8^{(3,4)} = 1$

$P_8^{(1,2)} = 1$

COMMUNICATION

- **Communication move**

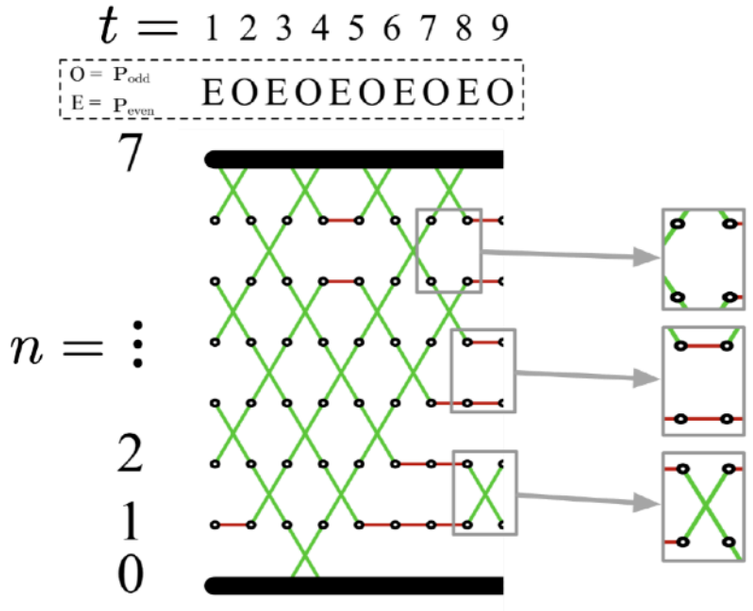
$$\mathbf{K}_{\text{comm}} = \prod_{n=1}^{N} K^{(n,n+1)} P_t^{(n,n+1)}$$

  - $P_t^{(n,n+1)} \in \{0,1\}$ indicated if the n-th swap is proposed
  - Note that in general, the order matters

- A special is odd and even swaps

$$\mathbf{K}_{\text{comm}} \in \{\mathbf{K}_{\text{odd}}, \mathbf{K}_{\text{even}}\}, \qquad \mathbf{K}_{\text{odd}} = \prod_{n \text{ odd}} \mathbf{K}^{(n,n+1)}, \quad \mathbf{K}_{\text{even}} = \prod_{n \text{ even}} \mathbf{K}^{(n,n+1)}$$

  - These are the largest collection of swap moves in parallel



Proposal indicators

$P_7^{(5,6)} = 0$

$P_8^{(3,4)} = 1$

$P_8^{(1,2)} = 1$

COMMUNICATION

- **Communication move**

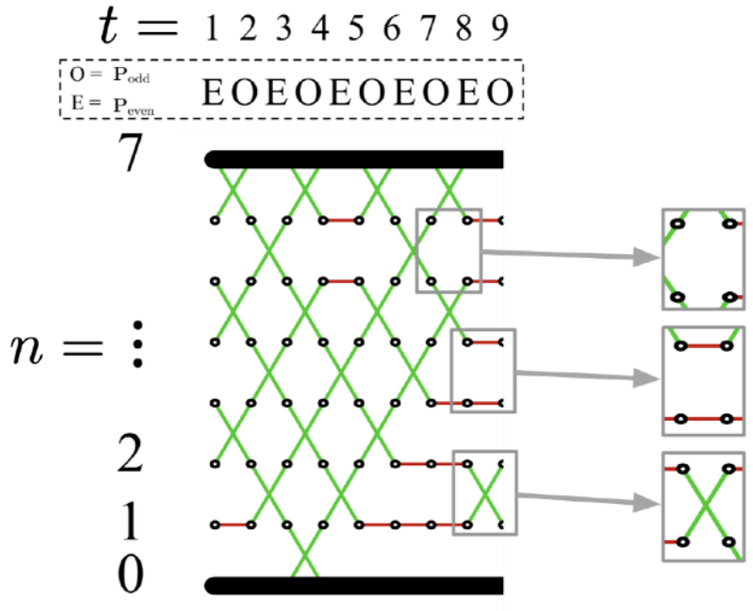
$$\mathbf{K}_{\text{comm}} = \prod_{n=1}^{N} K^{(n,n+1)} P_t^{(n,n+1)}$$

  - $P_t^{(n,n+1)} \in \{0,1\}$ indicated if the n-th swap is proposed
  - Note that in general, the order matters

- A special is odd and even swaps

$$\mathbf{K}_{\text{comm}} \in \{\mathbf{K}_{\text{odd}}, \mathbf{K}_{\text{even}}\}, \qquad \mathbf{K}_{\text{odd}} = \prod_{n \text{ odd}} \mathbf{K}^{(n,n+1)}, \quad \mathbf{K}_{\text{even}} = \prod_{n \text{ even}} \mathbf{K}^{(n,n+1)}$$

  - These are the largest collection of swap moves in parallel
  - Order does not matter!



Proposal indicators

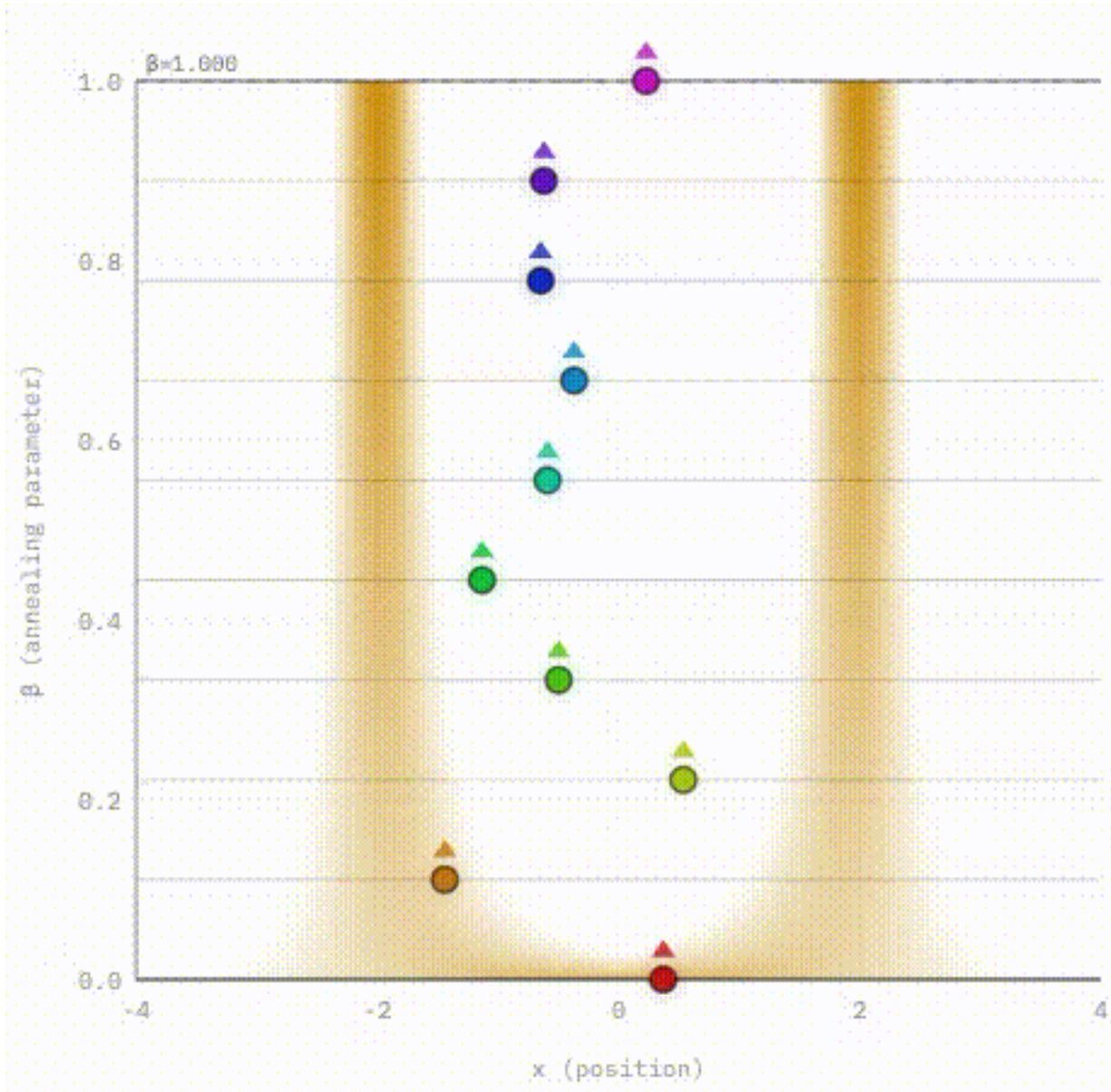
$P_7^{(5,6)} = 0$

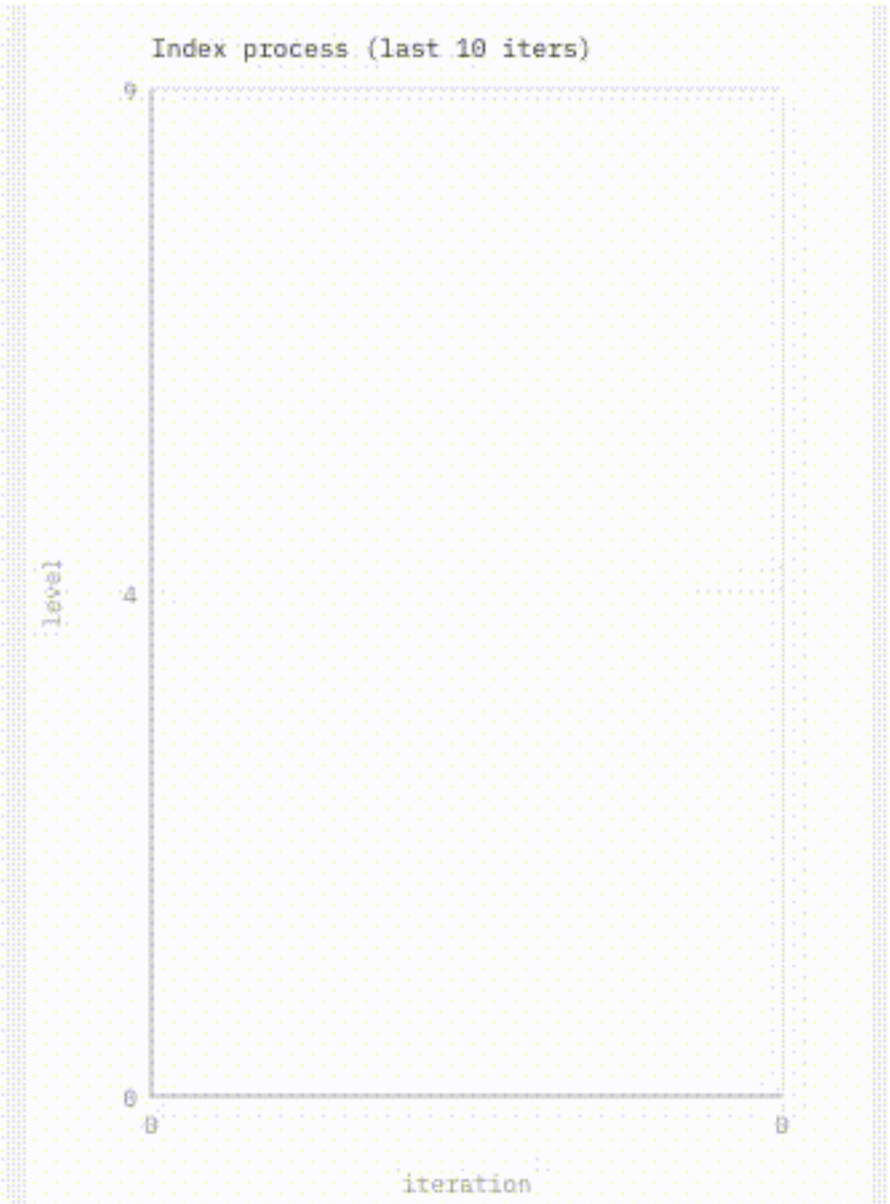
$P_8^{(3,4)} = 1$

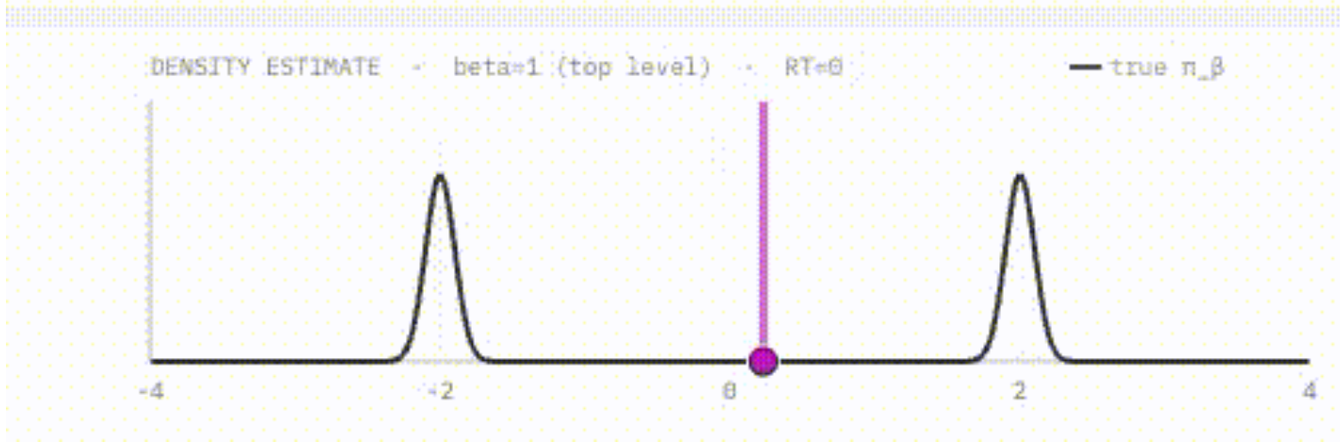
$P_8^{(1,2)} = 1$

PARALLEL TEMPERING

PARALLEL TEMPERING



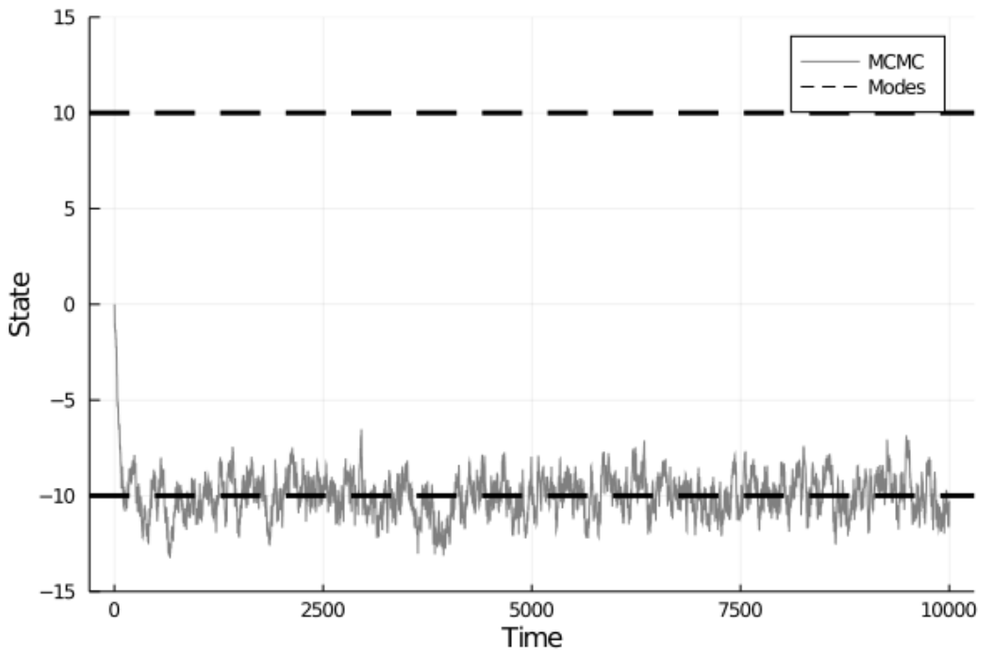


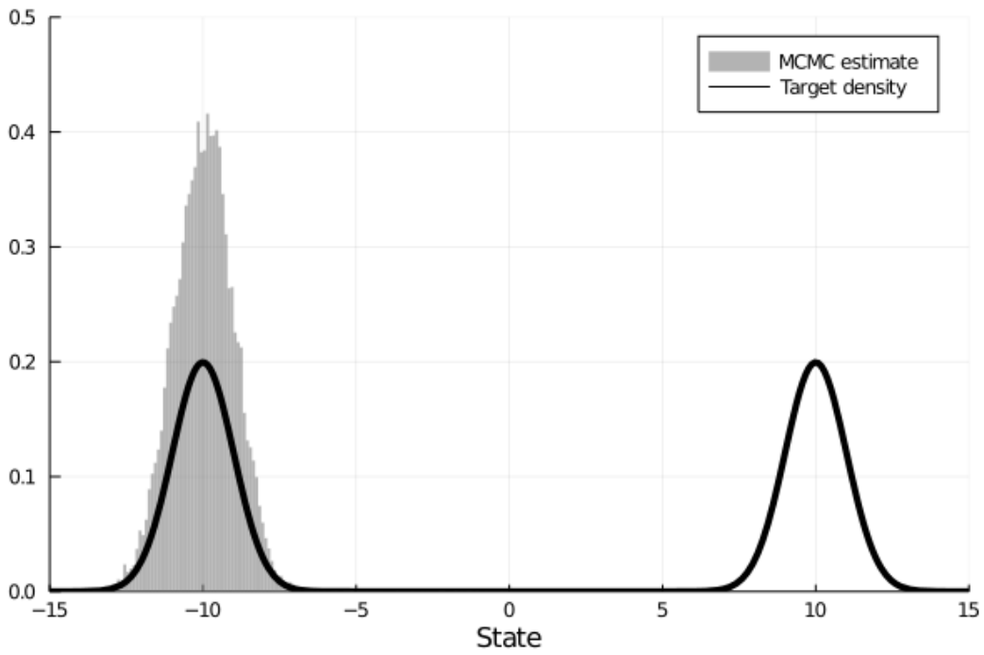


MCMC VS PARALLEL TEMPERING

MCMC VS PARALLEL TEMPERING

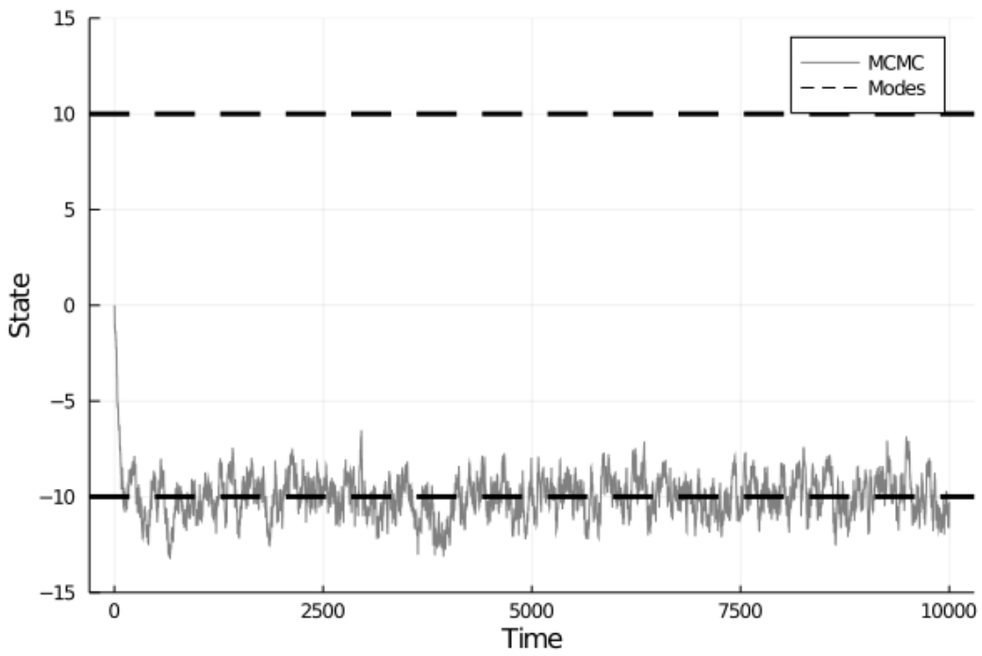
Single Chain MCMC

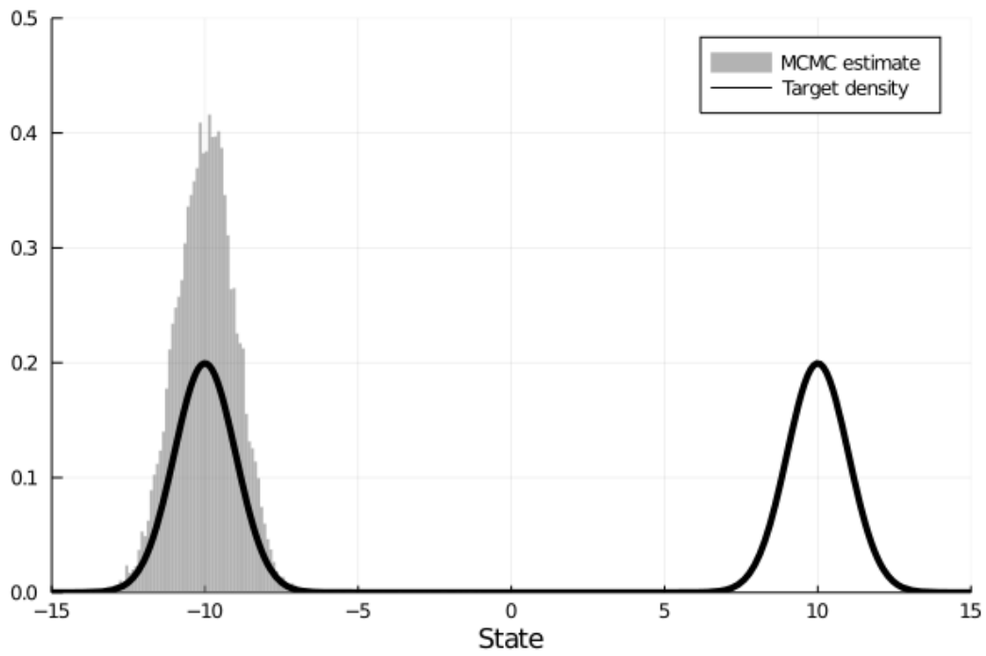




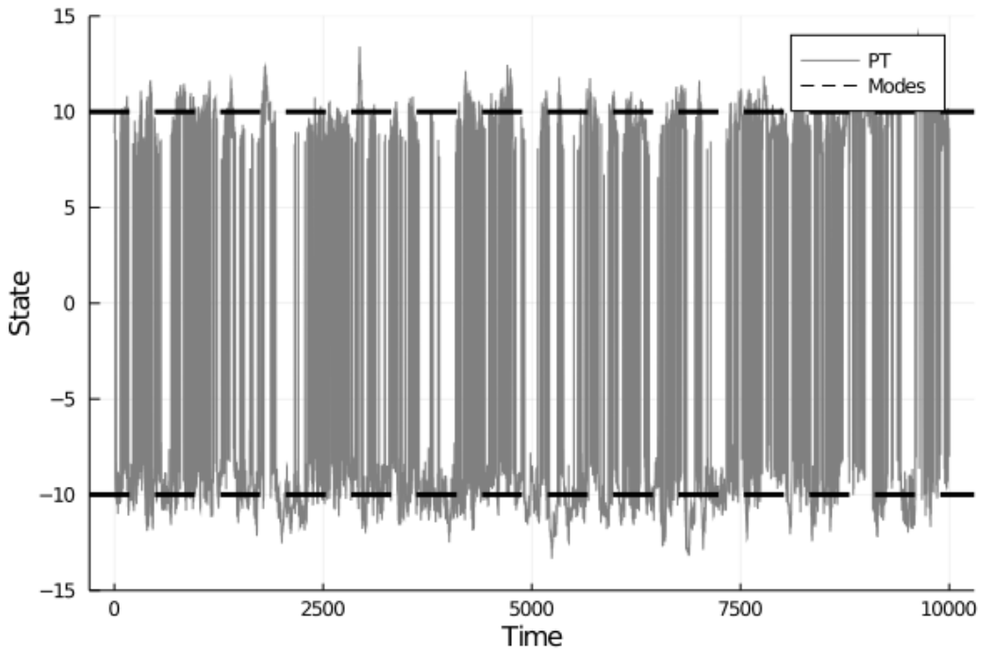
MCMC VS PARALLEL TEMPERING

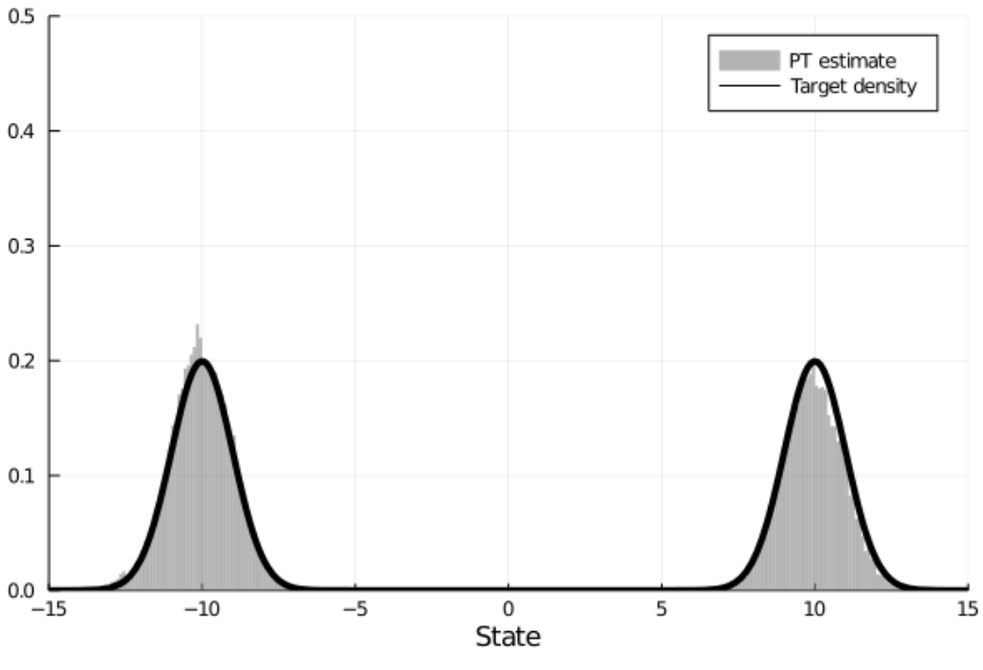
Single Chain MCMC





Parallel Tempering (N = 10)

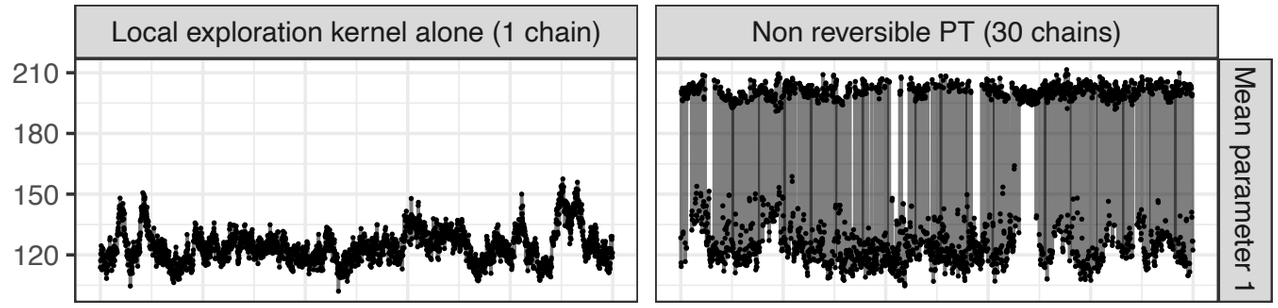




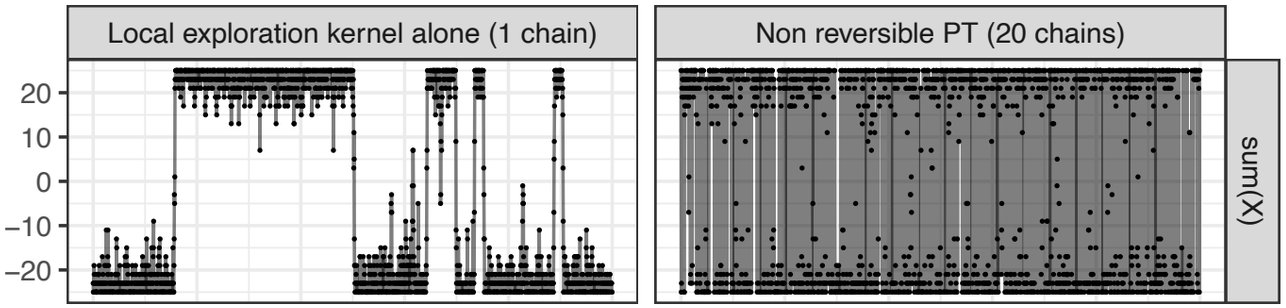
EXAMPLE: MULTIMODAL TARGETS

Single Chain MCMC **Parallel Tempering**

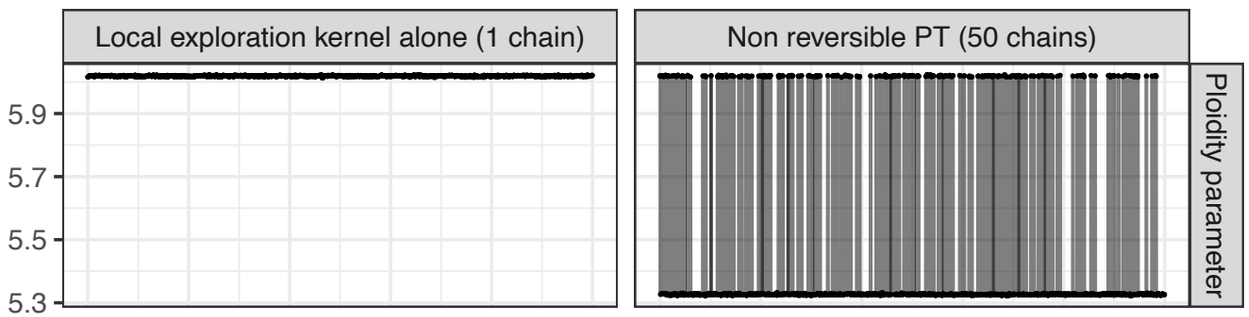
Bayesian Mixture Model
(d = 155)



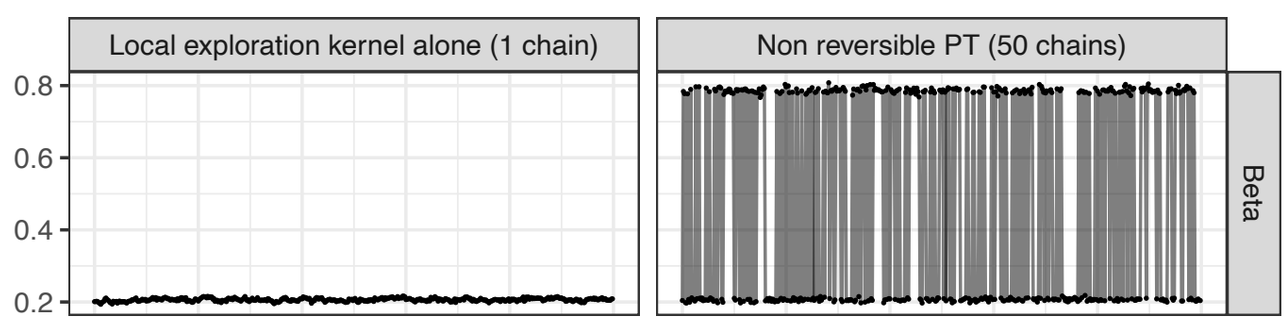
Ising Model
(d = 25)



Copy number inference
w/ whole genome ovarian cancer data
(d = 30)



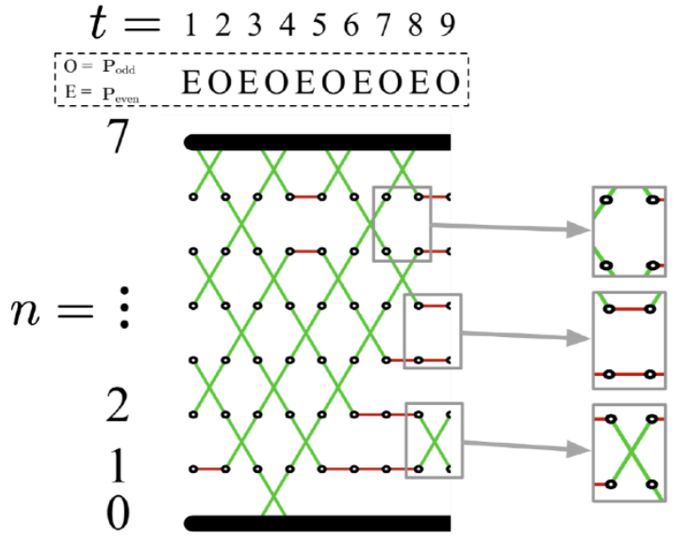
ODE parameter inference
w/ mRNA data
(d = 9)



DISTRIBUTED PT

DISTRIBUTED PT

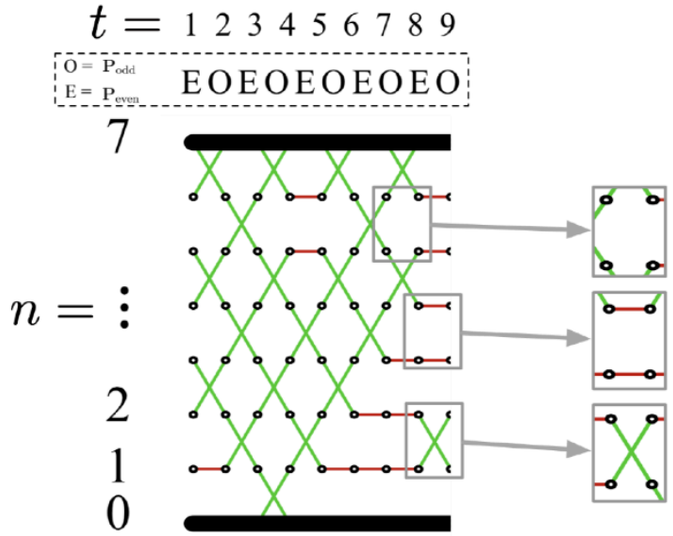
- **Dual perspective on PT:** Swap annealing parameters not states!



Proposal indicators	Acceptance indicator	Swap indicator
$P_7^{(5,6)} = 0$	$A_7^{(5,6)} = 1$	$S_7^{(5,6)} = 0$
$P_8^{(3,4)} = 1$	$A_8^{(3,4)} = 0$	$S_8^{(3,4)} = 0$
$P_8^{(1,2)} = 1$	$A_8^{(1,2)} = 1$	$S_8^{(1,2)} = 1$

DISTRIBUTED PT

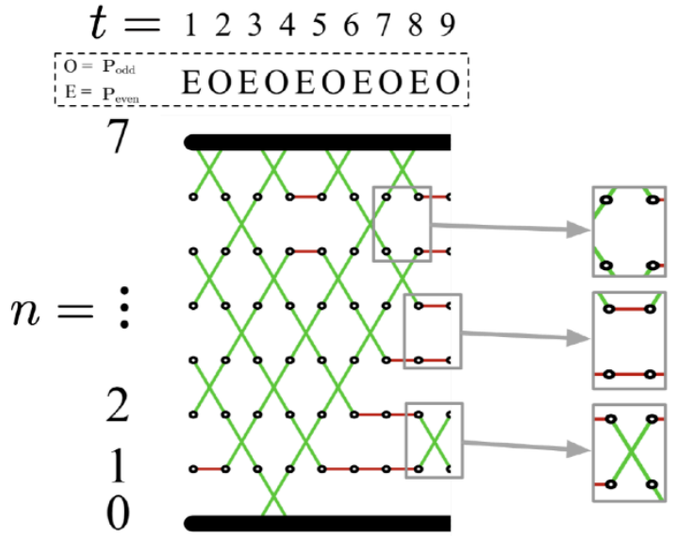
- **Dual perspective on PT:** Swap annealing parameters not states!
  - We can study communication through the dynamics of the permuted annealing parameters



	Proposal indicators	Acceptance indicator	Swap indicator
	$P_7^{(5,6)} = 0$	$A_7^{(5,6)} = 1$	$S_7^{(5,6)} = 0$
	$P_8^{(3,4)} = 1$	$A_8^{(3,4)} = 0$	$S_8^{(3,4)} = 0$
	$P_8^{(1,2)} = 1$	$A_8^{(1,2)} = 1$	$S_8^{(1,2)} = 1$

DISTRIBUTED PT

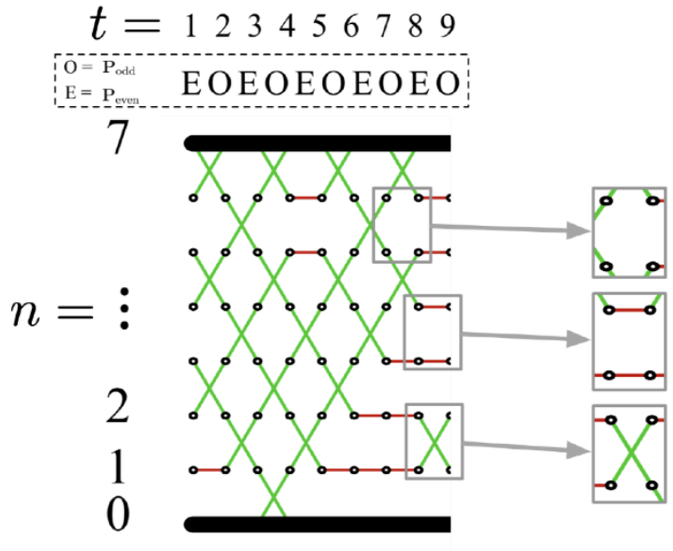
- **Dual perspective on PT:** Swap annealing parameters not states!
  - We can study communication through the dynamics of the permuted annealing parameters
  - More efficient for distributed implementation to swap numbers



	Proposal indicators	Acceptance indicator	Swap indicator
	$P_7^{(5,6)} = 0$	$A_7^{(5,6)} = 1$	$S_7^{(5,6)} = 0$
	$P_8^{(3,4)} = 1$	$A_8^{(3,4)} = 0$	$S_8^{(3,4)} = 0$
	$P_8^{(1,2)} = 1$	$A_8^{(1,2)} = 1$	$S_8^{(1,2)} = 1$

DISTRIBUTED PT

- **Dual perspective on PT:** Swap annealing parameters not states!
  - We can study communication through the dynamics of the permuted annealing parameters
  - More efficient for distributed implementation to swap numbers
- Machine m stores:
  - State: $Y_t^m \in \mathbb{X}$
  - Annealing parameter: $\beta_t^m = \beta_{I_t^m} \in \{\beta_0, \ldots, \beta_N\}$
  - Proposal direction: $\epsilon_t^m \in \{-1, 1\}$



	Proposal indicators	Acceptance indicator	Swap indicator
	$P_7^{(5,6)} = 0$	$A_7^{(5,6)} = 1$	$S_7^{(5,6)} = 0$
	$P_8^{(3,4)} = 1$	$A_8^{(3,4)} = 0$	$S_8^{(3,4)} = 0$
	$P_8^{(1,2)} = 1$	$A_8^{(1,2)} = 1$	$S_8^{(1,2)} = 1$

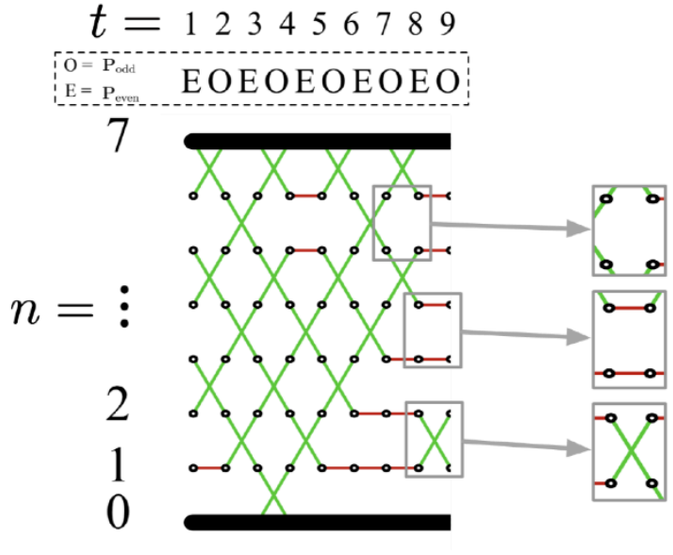
DISTRIBUTED PT

- **Dual perspective on PT:** Swap annealing parameters not states!
  - We can study communication through the dynamics of the permuted annealing parameters
  - More efficient for distributed implementation to swap numbers
- Machine m stores:
  - State: $Y_t^m \in \mathbb{X}$
  - Annealing parameter: $\beta_t^m = \beta_{I_t^m} \in \{\beta_0, \ldots, \beta_N\}$
  - Proposal direction: $\epsilon_t^m \in \{-1, 1\}$
- Distributed PT state:

$$\mathbf{Y}_t = (Y_t^0, \ldots, Y_t^N)$$

$$\mathbf{I}_t = (I_t^0, \ldots, I_t^N)$$

$$\boldsymbol{\epsilon}_t = (\epsilon_t^0, \ldots, \epsilon_t^N)$$



Proposal indicators	Acceptance indicator	Swap indicator
$P_7^{(5,6)} = 0$	$A_7^{(5,6)} = 1$	$S_7^{(5,6)} = 0$
$P_8^{(3,4)} = 1$	$A_8^{(3,4)} = 0$	$S_8^{(3,4)} = 0$
$P_8^{(1,2)} = 1$	$A_8^{(1,2)} = 1$	$S_8^{(1,2)} = 1$

DISTRIBUTED PT

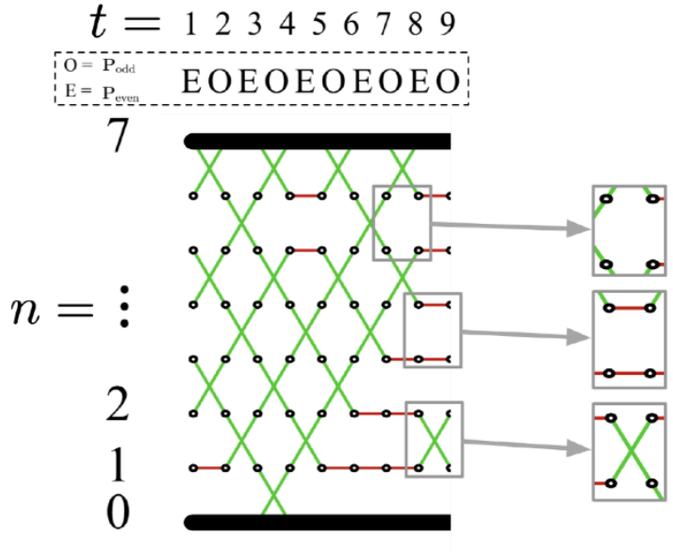
- **Dual perspective on PT:** Swap annealing parameters not states!
  - We can study communication through the dynamics of the permuted annealing parameters
  - More efficient for distributed implementation to swap numbers
- Machine m stores:
  - State: $Y_t^m \in \mathbb{X}$
  - Annealing parameter: $\beta_t^m = \beta_{I_t^m} \in \{\beta_0, \ldots, \beta_N\}$
  - Proposal direction: $\epsilon_t^m \in \{-1, 1\}$
- Distributed PT state:

$$\mathbf{Y}_t = (Y_t^0, \ldots, Y_t^N)$$

$$\mathbf{I}_t = (I_t^0, \ldots, I_t^N)$$

$$\boldsymbol{\epsilon}_t = (\epsilon_t^0, \ldots, \epsilon_t^N)$$

- X_t^n is commonly referred to as n-th **chain** and Y_t^m the m-th **replica**



	Proposal indicators	Acceptance indicator	Swap indicator
	$P_7^{(5,6)} = 0$	$A_7^{(5,6)} = 1$	$S_7^{(5,6)} = 0$
	$P_8^{(3,4)} = 1$	$A_8^{(3,4)} = 0$	$S_8^{(3,4)} = 0$
	$P_8^{(1,2)} = 1$	$A_8^{(1,2)} = 1$	$S_8^{(1,2)} = 1$

DISTRIBUTED PT

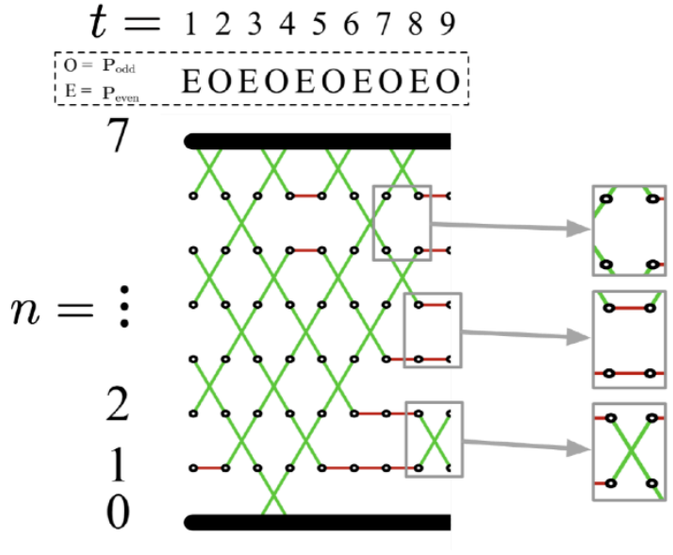
- **Dual perspective on PT:** Swap annealing parameters not states!
  - We can study communication through the dynamics of the permuted annealing parameters
  - More efficient for distributed implementation to swap numbers
- Machine m stores:
  - State: $Y_t^m \in \mathbb{X}$
  - Annealing parameter: $\beta_t^m = \beta_{I_t^m} \in \{\beta_0, \ldots, \beta_N\}$
  - Proposal direction: $\epsilon_t^m \in \{-1, 1\}$
- Distributed PT state:

$$\mathbf{Y}_t = (Y_t^0, \ldots, Y_t^N)$$

$$\mathbf{I}_t = (I_t^0, \ldots, I_t^N)$$

$$\boldsymbol{\epsilon}_t = (\epsilon_t^0, \ldots, \epsilon_t^N)$$

- X_t^n is commonly referred to as n-th **chain** and Y_t^m the m-th **replica**
- We note that $\mathbf{Y}_t$ is $\mathbf{X}_t$ shuffled according to permutation $\mathbf{I}_t$



Proposal indicators	Acceptance indicator	Swap indicator
$P_7^{(5,6)} = 0$	$A_7^{(5,6)} = 1$	$S_7^{(5,6)} = 0$
$P_8^{(3,4)} = 1$	$A_8^{(3,4)} = 0$	$S_8^{(3,4)} = 0$
$P_8^{(1,2)} = 1$	$A_8^{(1,2)} = 1$	$S_8^{(1,2)} = 1$

DISTRIBUTED PT

DISTRIBUTED PT

- **Intialise:**

$$Y_0^m = X_0^m, \qquad I_0^m = m, \qquad \epsilon_0^m = (-1)^m$$

DISTRIBUTED PT

- **Intialise:**

$$Y_0^m = X_0^m, \qquad I_0^m = m, \qquad \epsilon_0^m = (-1)^m$$

- **Local exploration:** For each m update Y_t^m with the local move $K_{\beta_t^m}$ targeting $\pi_{\beta_t^m}$

$$Y_{t+1}^m \sim K_{\beta_t^m}(Y_t^m, \mathrm{d}y)$$

DISTRIBUTED PT

- **Intialise:**

$$Y_0^m = X_0^m, \qquad I_0^m = m, \qquad \epsilon_0^m = (-1)^m$$

- **Local exploration:** For each m update Y_t^m with the local move $K_{\beta_t^m}$ targeting $\pi_{\beta_t^m}$

$$Y_{t+1}^m \sim K_{\beta_t^m}(Y_t^m, \mathrm{d}y)$$

- **Communication:** machine m proposes swaps of indexes I_{t-1}^m with machine storing $I_{t-1}^m + \epsilon_{t-1}^m$

DISTRIBUTED PT

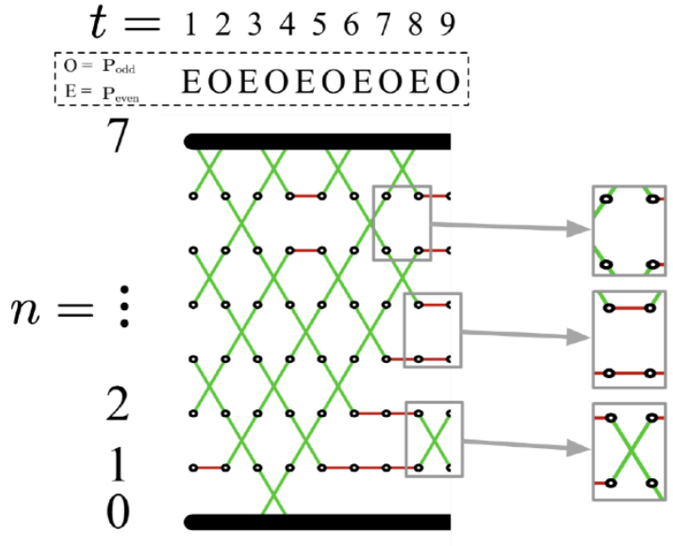
- **Intialise:**

$$Y_0^m = X_0^m, \qquad I_0^m = m, \qquad \epsilon_0^m = (-1)^m$$

- **Local exploration:** For each m update Y_t^m with the local move $K_{\beta_t^m}$ targeting $\pi_{\beta_t^m}$

$$Y_{t+1}^m \sim K_{\beta_t^m}(Y_t^m, \mathrm{d}y)$$

- **Communication:** machine m proposes swaps of indexes I_{t-1}^m with machine storing $I_{t-1}^m + \epsilon_{t-1}^m$



Proposal indicators	Acceptance indicator	Swap indicator
$P_7^{(5,6)} = 0$	$A_7^{(5,6)} = 1$	$S_7^{(5,6)} = 0$
$P_8^{(3,4)} = 1$	$A_8^{(3,4)} = 0$	$S_8^{(3,4)} = 0$
$P_8^{(1,2)} = 1$	$A_8^{(1,2)} = 1$	$S_8^{(1,2)} = 1$

DISTRIBUTED PT

- **Intialise:**

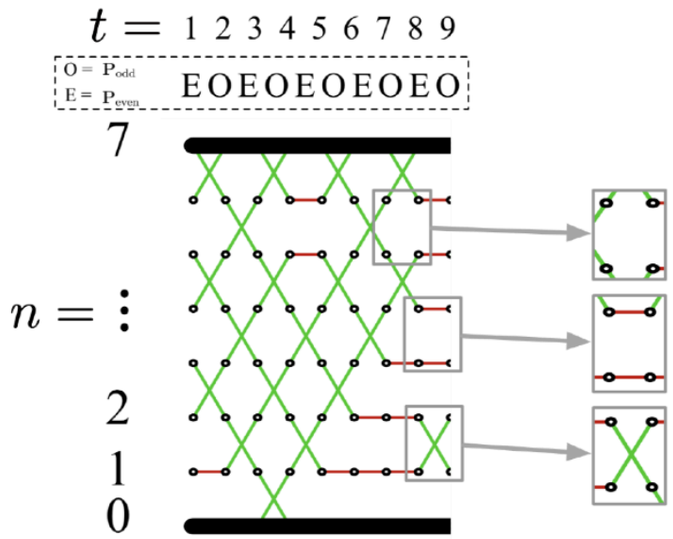
$$Y_0^m = X_0^m, \qquad I_0^m = m, \qquad \epsilon_0^m = (-1)^m$$

- **Local exploration:** For each m update Y_t^m with the local move $K_{\beta_t^m}$ targeting $\pi_{\beta_t^m}$

$$Y_{t+1}^m \sim K_{\beta_t^m}(Y_t^m, \mathrm{d}y)$$

- **Communication:** machine m proposes swaps of indexes I_{t-1}^m with machine storing $I_{t-1}^m + \epsilon_{t-1}^m$
  - If proposed swap is accepted:

$$I_{t+1}^m = I_t^m + \epsilon_t^m, \qquad S_t^{(I_t^m, I_t^m + \epsilon_t^m)} = 1$$



	Proposal indicators	Acceptance indicator	Swap indicator
	$P_7^{(5,6)} = 0$	$A_7^{(5,6)} = 1$	$S_7^{(5,6)} = 0$
	$P_8^{(3,4)} = 1$	$A_8^{(3,4)} = 0$	$S_8^{(3,4)} = 0$
	$P_8^{(1,2)} = 1$	$A_8^{(1,2)} = 1$	$S_8^{(1,2)} = 1$

DISTRIBUTED PT

- **Intialise:**

$$Y_0^m = X_0^m, \qquad I_0^m = m, \qquad \epsilon_0^m = (-1)^m$$

- **Local exploration:** For each m update Y_t^m with the local move $K_{\beta_t^m}$ targeting $\pi_{\beta_t^m}$

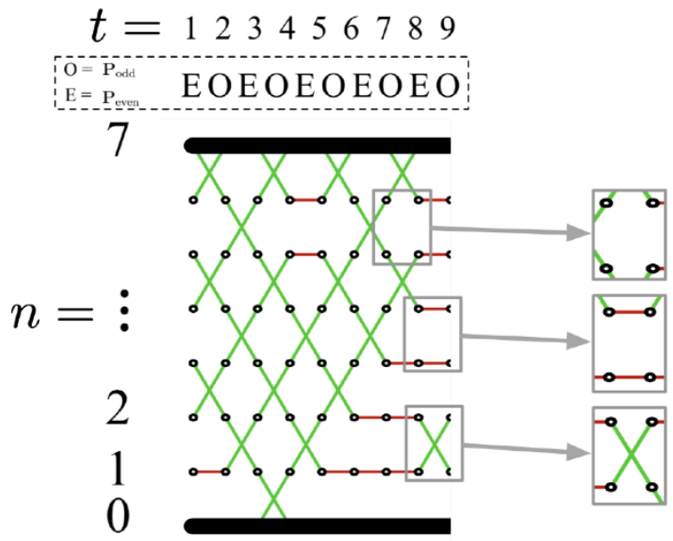
$$Y_{t+1}^m \sim K_{\beta_t^m}(Y_t^m, \mathrm{d}y)$$

- **Communication:** machine m proposes swaps of indexes I_{t-1}^m with machine storing $I_{t-1}^m + \epsilon_{t-1}^m$
  - If proposed swap is accepted:

$$I_{t+1}^m = I_t^m + \epsilon_t^m, \qquad S_t^{(I_t^m, I_t^m + \epsilon_t^m)} = 1$$

  - Otherwise

$$I_{t+1}^m = I_t^m$$



	Proposal indicators	Acceptance indicator	Swap indicator
	$P_7^{(5,6)} = 0$	$A_7^{(5,6)} = 1$	$S_7^{(5,6)} = 0$
	$P_8^{(3,4)} = 1$	$A_8^{(3,4)} = 0$	$S_8^{(3,4)} = 0$
	$P_8^{(1,2)} = 1$	$A_8^{(1,2)} = 1$	$S_8^{(1,2)} = 1$

DISTRIBUTED PT

- **Intialise:**

$$Y_0^m = X_0^m, \qquad I_0^m = m, \qquad \epsilon_0^m = (-1)^m$$

- **Local exploration:** For each m update Y_t^m with the local move $K_{\beta_t^m}$ targeting $\pi_{\beta_t^m}$

$$Y_{t+1}^m \sim K_{\beta_t^m}(Y_t^m, \mathrm{d}y)$$

- **Communication:** machine m proposes swaps of indexes I_{t-1}^m with machine storing $I_{t-1}^m + \epsilon_{t-1}^m$
  - If proposed swap is accepted:

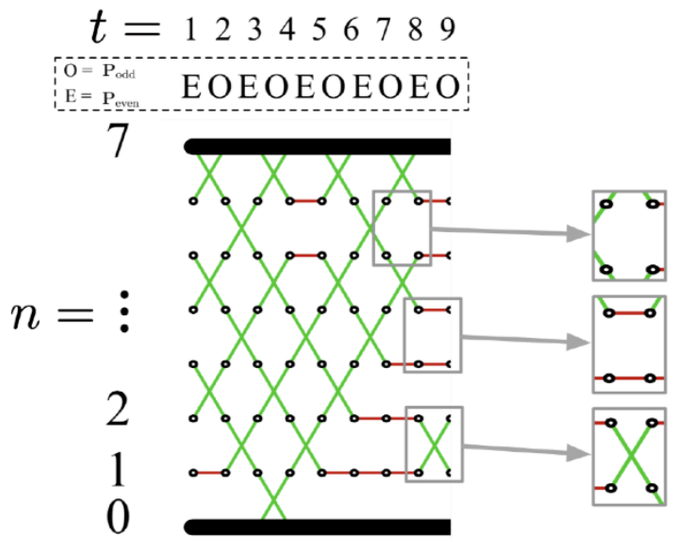
$$I_{t+1}^m = I_t^m + \epsilon_t^m, \qquad S_t^{(I_t^m, I_t^m + \epsilon_t^m)} = 1$$

  - If the next scheduled proposal is an increase

$$\epsilon_t = 1, \qquad P_{t+1}^{(I_{t+1}^m, I_{t+1}^m + 1)} = 1$$

  - Otherwise

$$I_{t+1}^m = I_t^m$$



Proposal indicators	Acceptance indicator	Swap indicator
$P_7^{(5,6)} = 0$	$A_7^{(5,6)} = 1$	$S_7^{(5,6)} = 0$
$P_8^{(3,4)} = 1$	$A_8^{(3,4)} = 0$	$S_8^{(3,4)} = 0$
$P_8^{(1,2)} = 1$	$A_8^{(1,2)} = 1$	$S_8^{(1,2)} = 1$

DISTRIBUTED PT

- **Intialise:**

$$Y_0^m = X_0^m, \qquad I_0^m = m, \qquad \epsilon_0^m = (-1)^m$$

- **Local exploration:** For each m update Y_t^m with the local move $K_{\beta_t^m}$ targeting $\pi_{\beta_t^m}$

$$Y_{t+1}^m \sim K_{\beta_t^m}(Y_t^m, \mathrm{d}y)$$

- **Communication:** machine m proposes swaps of indexes I_{t-1}^m with machine storing $I_{t-1}^m + \epsilon_{t-1}^m$
  - If proposed swap is accepted:

$$I_{t+1}^m = I_t^m + \epsilon_t^m, \qquad S_t^{(I_t^m, I_t^m + \epsilon_t^m)} = 1$$

  - Otherwise

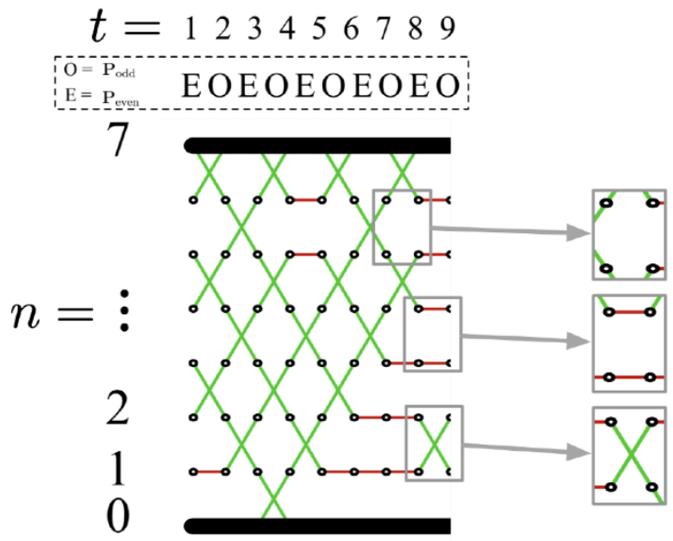
$$I_{t+1}^m = I_t^m$$

  - If the next scheduled proposal is an increase

$$\epsilon_t = 1, \qquad P_{t+1}^{(I_{t+1}^m, I_{t+1}^m + 1)} = 1$$

  - Otherwise

$$\epsilon_t = -1$$



Proposal indicators	Acceptance indicator	Swap indicator
$P_7^{(5,6)} = 0$	$A_7^{(5,6)} = 1$	$S_7^{(5,6)} = 0$
$P_8^{(3,4)} = 1$	$A_8^{(3,4)} = 0$	$S_8^{(3,4)} = 0$
$P_8^{(1,2)} = 1$	$A_8^{(1,2)} = 1$	$S_8^{(1,2)} = 1$

OBJECTIVE OF PT

OBJECTIVE OF PT

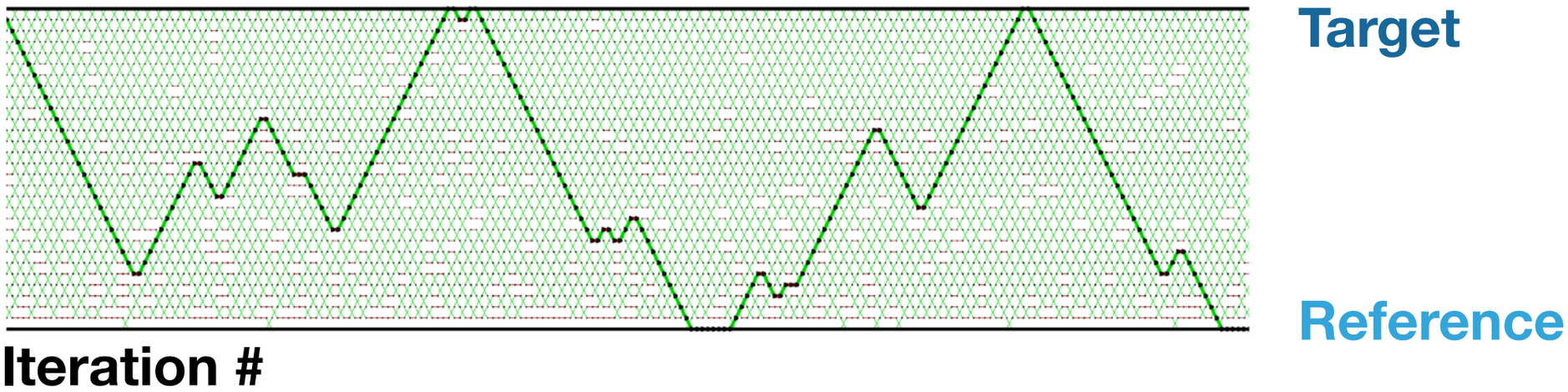
- We can study the communication through index processes I_t^m

OBJECTIVE OF PT

- We can study the communication through index processes I_t^m
- **Restart:** When I_t^m travels from 0 to N, reference and target have communicated on machine m

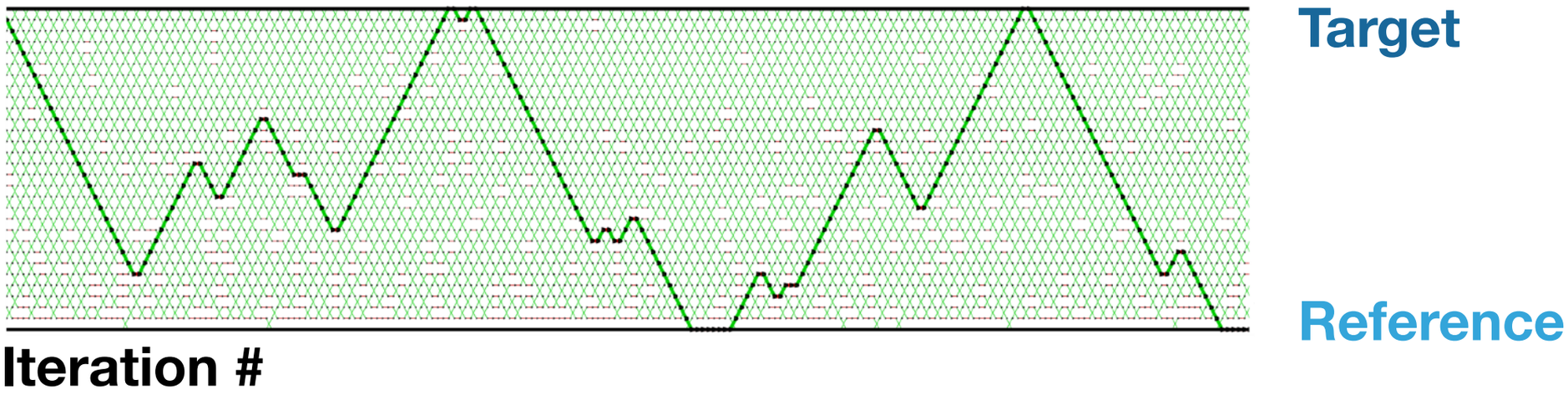
OBJECTIVE OF PT

- We can study the communication through index processes I_t^m
- **Restart:** When I_t^m travels from 0 to N, reference and target have communicated on machine m
- **Round Trip:** when a I_t^m travels from 0 to N and back to 0



OBJECTIVE OF PT

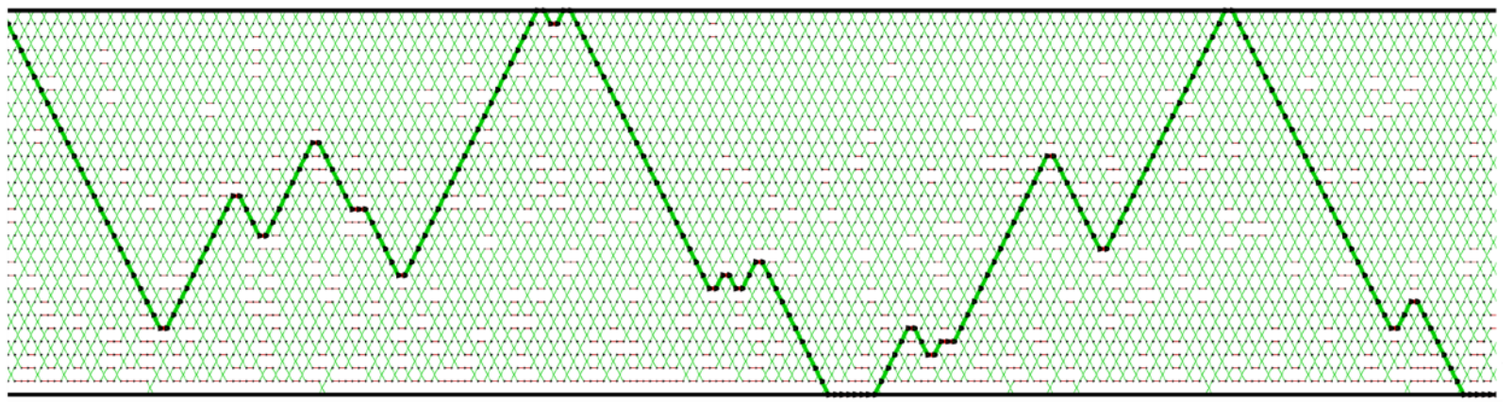
- We can study the communication through index processes I_t^m
- **Restart:** When I_t^m travels from 0 to N, reference and target have communicated on machine m
- **Round Trip:** when a I_t^m travels from 0 to N and back to 0



- Total round-trip and restarts differ by at most N and hence are equivalent to study

OBJECTIVE OF PT

- We can study the communication through index processes I_t^m
- **Restart:** When I_t^m travels from 0 to N, reference and target have communicated on machine m
- **Round Trip:** when a I_t^m travels from 0 to N and back to 0



- Total round-trip and restarts differ by at most N and hence are equivalent to study
- **Round trip rate** % of iterations with round trip

$$\tau_N = \lim_{t \to \infty} \frac{\mathbb{E}[\text{total round trips by time t}]}{t}$$

REVERSIBLE PT

REVERSIBLE PT

- Propose swaps at random in communication phase

REVERSIBLE PT

- Propose swaps at random in communication phase

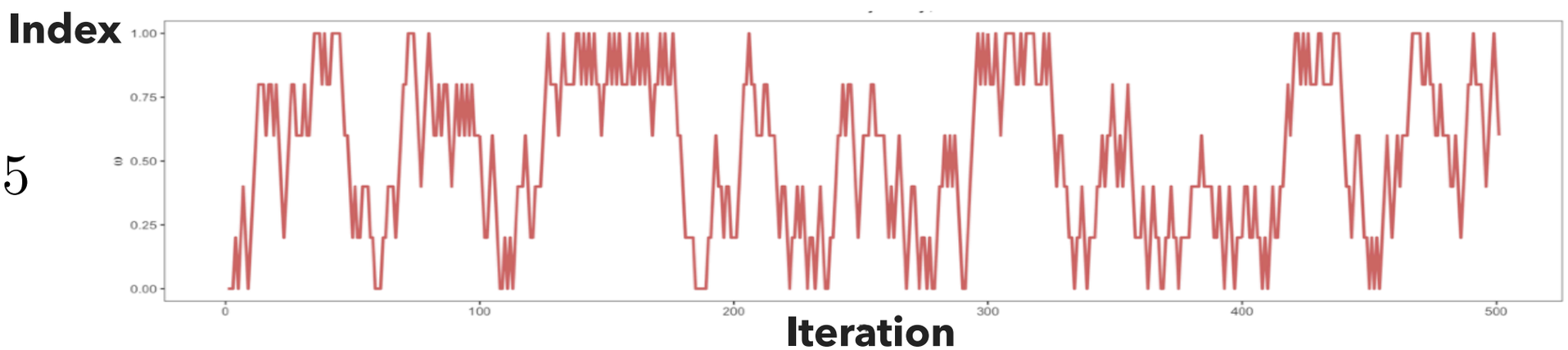
$$\mathbf{K}_{\text{comm}} = \frac{1}{2}\mathbf{K}_{\text{odd}} + \frac{1}{2}\mathbf{K}_{\text{even}}$$

REVERSIBLE PT

- Propose swaps at random in communication phase

$$\mathbf{K}_{\text{comm}} = \frac{1}{2}\mathbf{K}_{\text{odd}} + \frac{1}{2}\mathbf{K}_{\text{even}}$$

$N = 5$

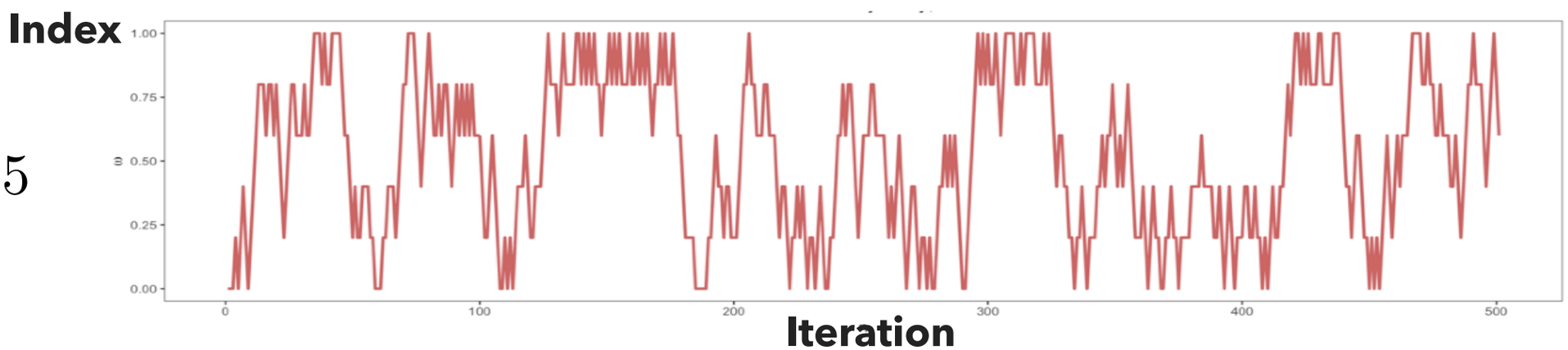


REVERSIBLE PT

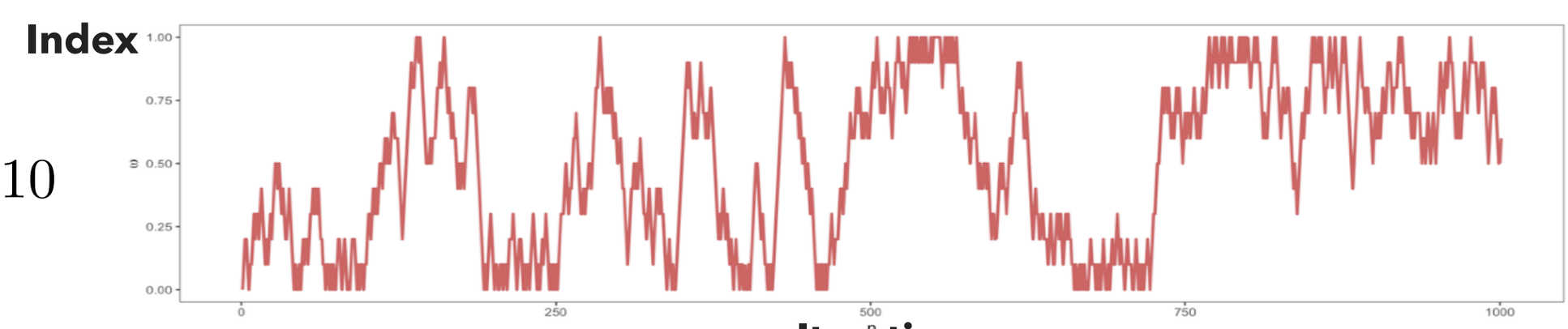
- Propose swaps at random in communication phase

$$\mathbf{K}_{\text{comm}} = \frac{1}{2}\mathbf{K}_{\text{odd}} + \frac{1}{2}\mathbf{K}_{\text{even}}$$

$N = 5$



$N = 10$

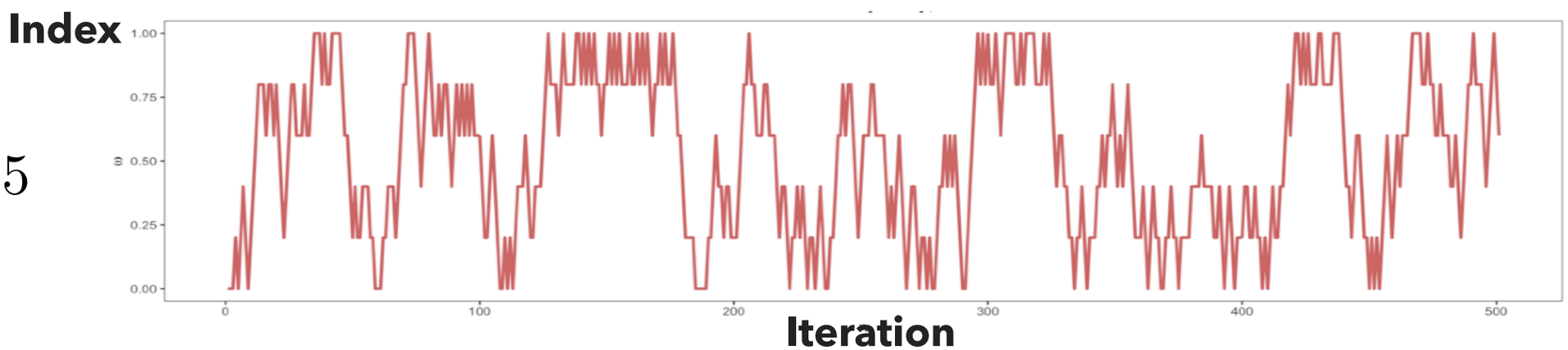


REVERSIBLE PT

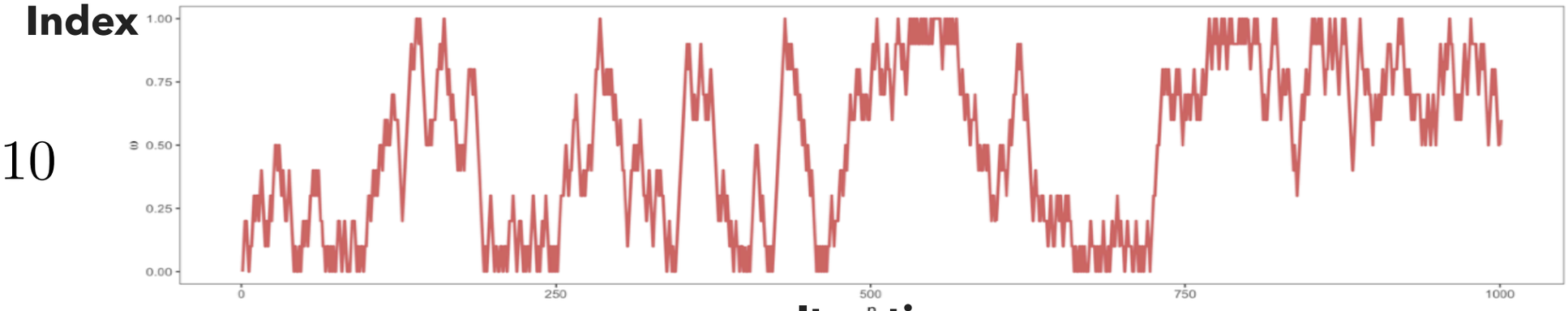
- Propose swaps at random in communication phase

$$\mathbf{K}_{\text{comm}} = \frac{1}{2}\mathbf{K}_{\text{odd}} + \frac{1}{2}\mathbf{K}_{\text{even}}$$

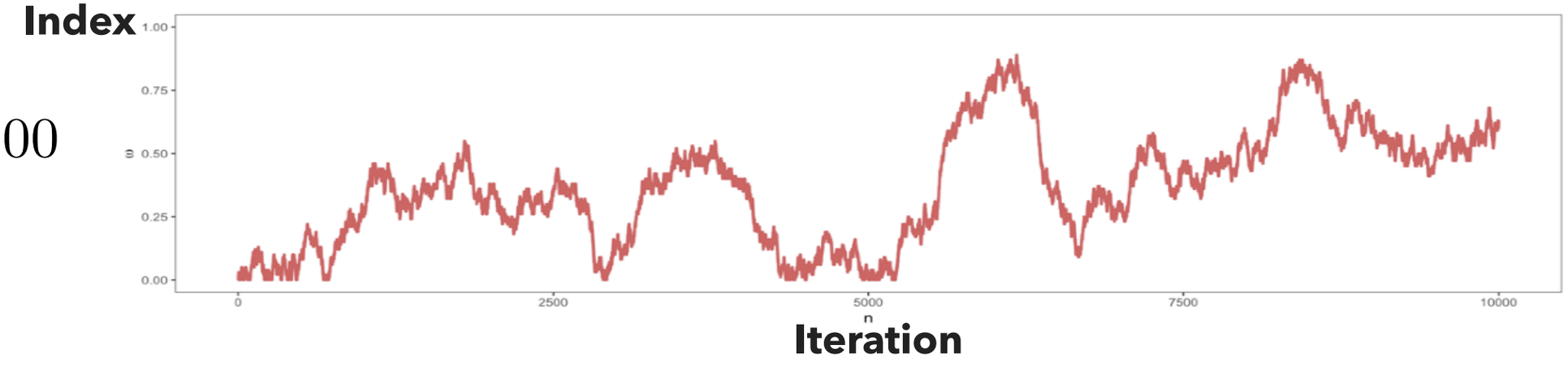
$N = 5$



$N = 10$



$N = 100$



REVERSIBLE PT

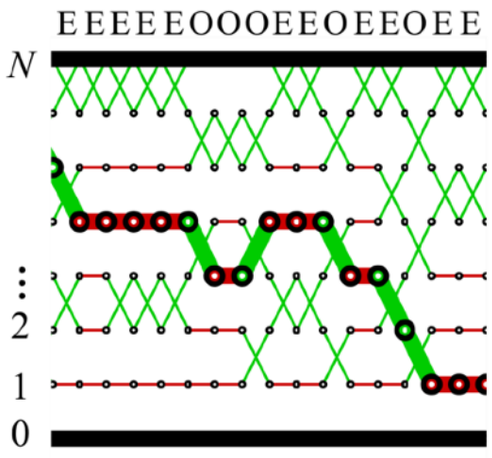
- Propose swaps at random in communication phase

$$\mathbf{K}_{\text{comm}} = \frac{1}{2}\mathbf{K}_{\text{odd}} + \frac{1}{2}\mathbf{K}_{\text{even}}$$

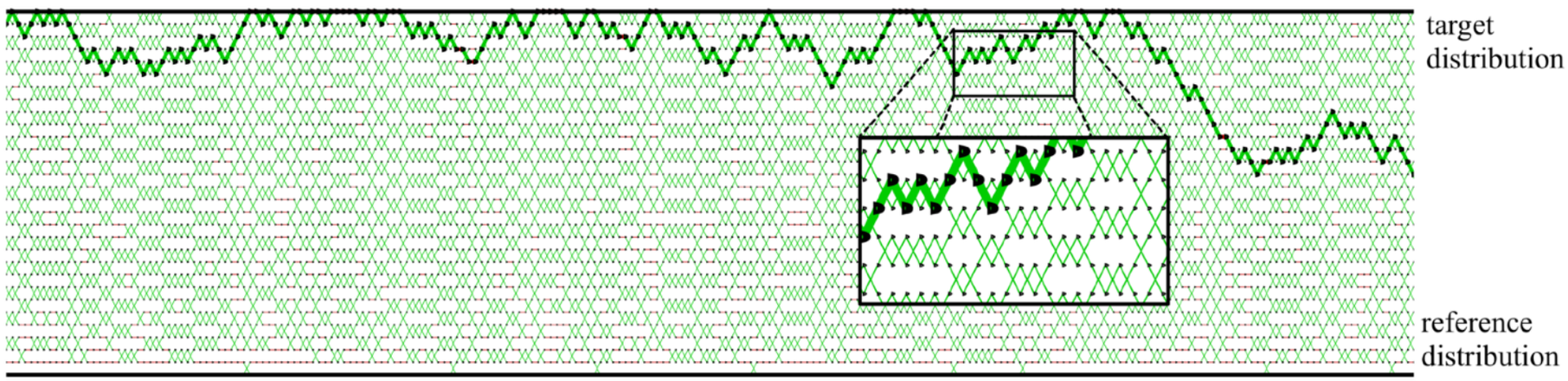
- Propose swaps at random in communication phase

$$\epsilon_{t+1}^{m} \sim \text{Uniform}\{-1,1\}$$

$N = 8$



$N = 30$



REVERSIBLE PT

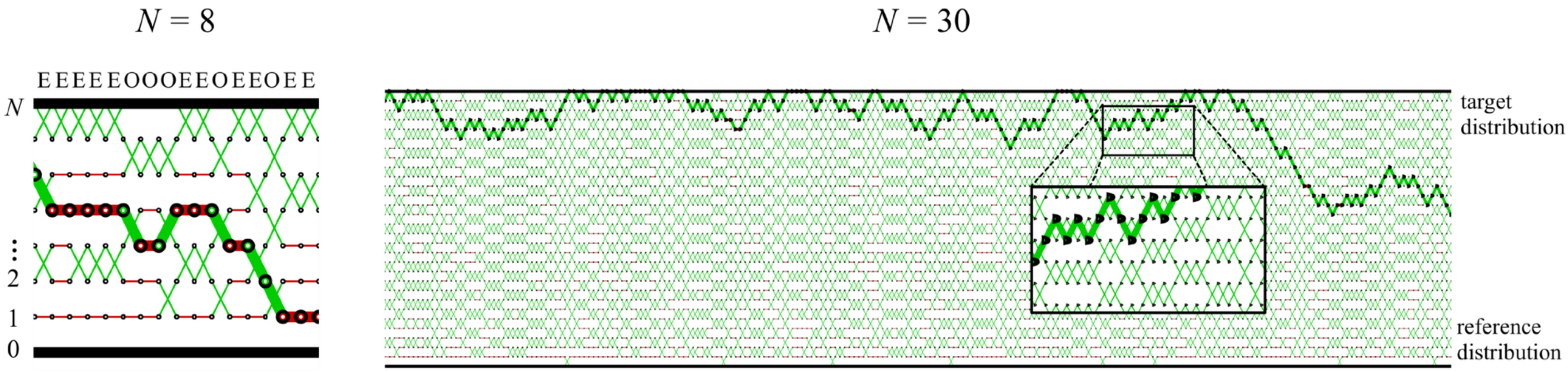
- Propose swaps at random in communication phase

$$\mathbf{K}_{\text{comm}} = \frac{1}{2}\mathbf{K}_{\text{odd}} + \frac{1}{2}\mathbf{K}_{\text{even}}$$

- Propose swaps at random in communication phase

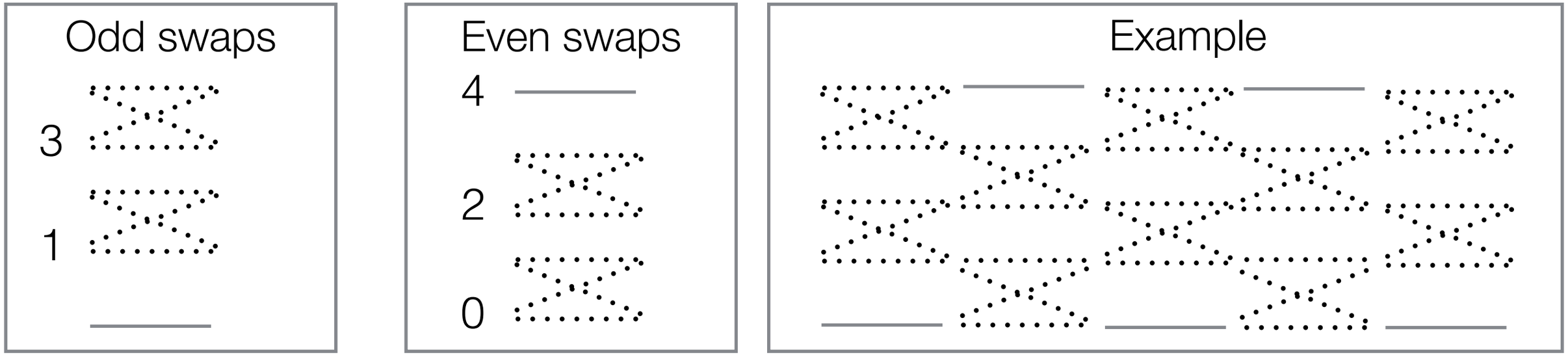
$$\epsilon_{t+1}^{m} \sim \text{Uniform}\{-1,1\}$$

- The index process behaves like a lazy random walk!



NON-REVERSIBLE PT (OKABE ET AL., 2001)

- Deterministically alternate between **Odd** and **Even** swaps

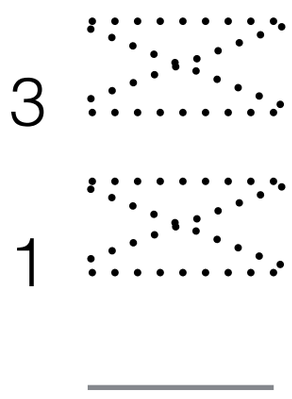


NON-REVERSIBLE PT (OKABE ET AL, 2001)

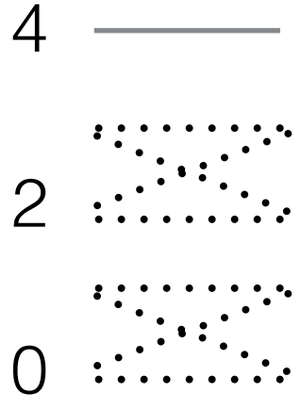
- Deterministically alternate between **Odd** and **Even** swaps

$$\mathbf{K}_{\text{comm},t} = \mathbf{K}_{\text{even}} 1(t \equiv 0 \mod 2) + \mathbf{K}_{\text{odd}} 1(t \equiv 1 \mod 2)$$

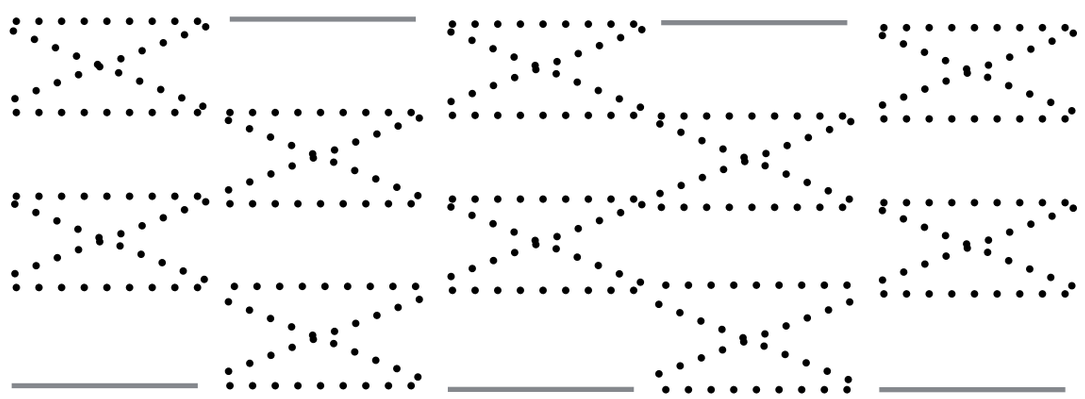
Odd swaps



Even swaps



Example



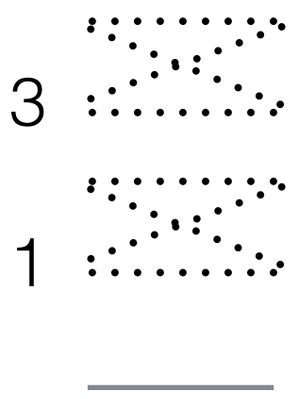
NON-REVERSIBLE PT (OKABE ET AL, 2001)

- Deterministically alternate between **Odd** and **Even** swaps

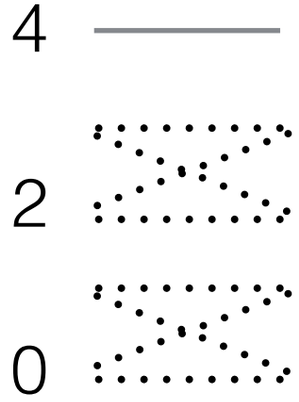
$$\mathbf{K}_{\text{comm},t} = \mathbf{K}_{\text{even}} 1(t \equiv 0 \mod 2) + \mathbf{K}_{\text{odd}} 1(t \equiv 1 \mod 2)$$

$$= \prod_{n \equiv t \mod 2} \mathbf{K}^{(n,n+1)}$$

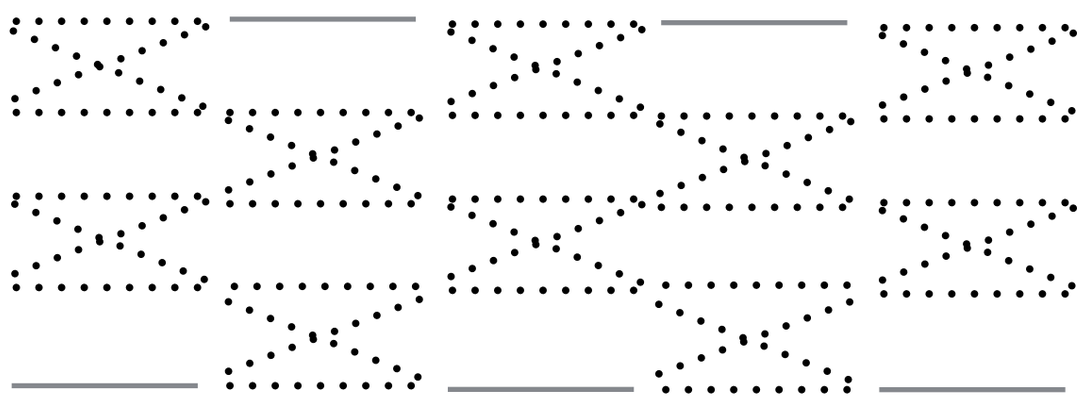
Odd swaps



Even swaps



Example

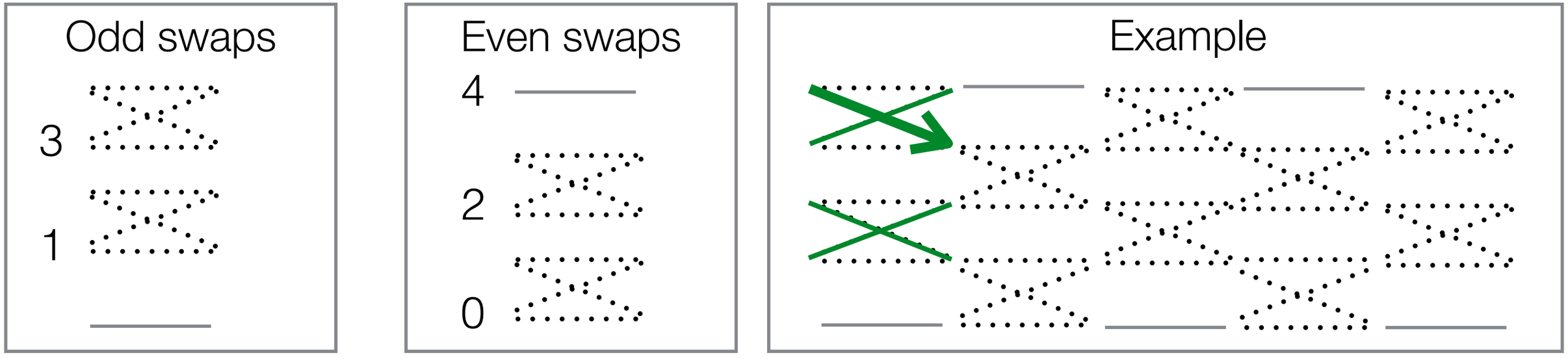


NON-REVERSIBLE PT (OKABE ET AL, 2001)

- Deterministically alternate between **Odd** and **Even** swaps

$$\mathbf{K}_{\text{comm},t} = \mathbf{K}_{\text{even}} 1(t \equiv 0 \mod 2) + \mathbf{K}_{\text{odd}} 1(t \equiv 1 \mod 2)$$

$$= \prod_{n \equiv t \mod 2} \mathbf{K}^{(n,n+1)}$$

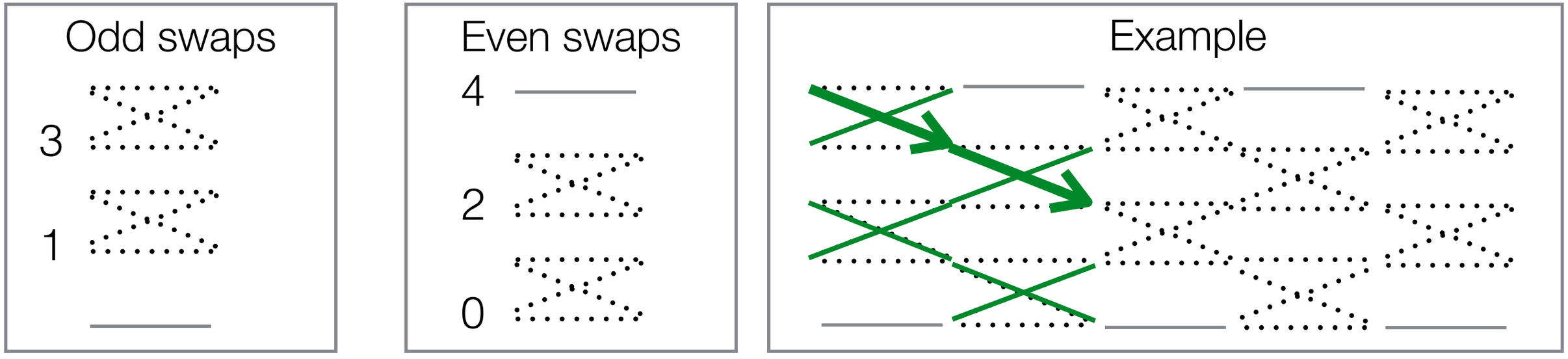


NON-REVERSIBLE PT (OKABE ET AL, 2001)

- Deterministically alternate between **Odd** and **Even** swaps

$$\mathbf{K}_{\text{comm},t} = \mathbf{K}_{\text{even}} 1(t \equiv 0 \mod 2) + \mathbf{K}_{\text{odd}} 1(t \equiv 1 \mod 2)$$

$$= \prod_{n \equiv t \mod 2} \mathbf{K}^{(n,n+1)}$$

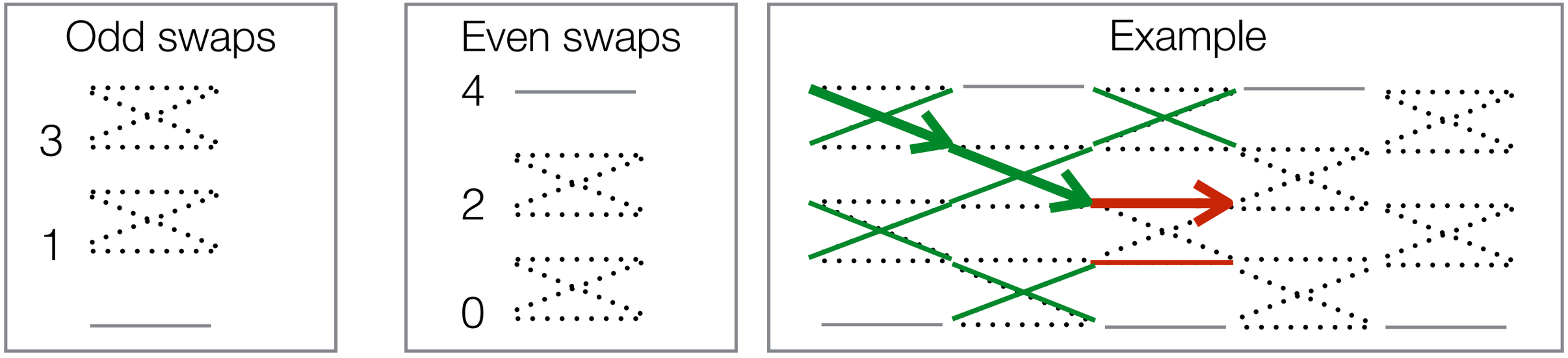


NON-REVERSIBLE PT (OKABE ET AL, 2001)

- Deterministically alternate between **Odd** and **Even** swaps

$$\mathbf{K}_{\text{comm},t} = \mathbf{K}_{\text{even}} 1(t \equiv 0 \mod 2) + \mathbf{K}_{\text{odd}} 1(t \equiv 1 \mod 2)$$

$$= \prod_{n \equiv t \mod 2} \mathbf{K}^{(n,n+1)}$$

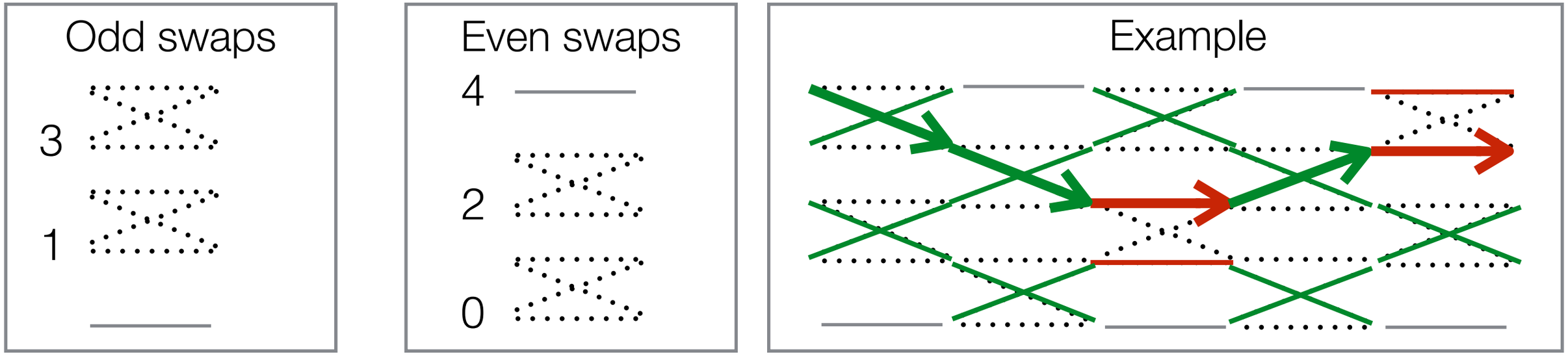


NON-REVERSIBLE PT (OKABE ET AL, 2001)

- Deterministically alternate between **Odd** and **Even** swaps

$$\mathbf{K}_{\text{comm},t} = \mathbf{K}_{\text{even}} 1(t \equiv 0 \mod 2) + \mathbf{K}_{\text{odd}} 1(t \equiv 1 \mod 2)$$

$$= \prod_{n \equiv t \mod 2} \mathbf{K}^{(n,n+1)}$$



NON-REVERSIBLE PT (OKABE ET AL, 2001)

- Deterministically alternate between **Odd** and **Even** swaps

$$\mathbf{K}_{\text{comm},t} = \mathbf{K}_{\text{even}} 1(t \equiv 0 \mod 2) + \mathbf{K}_{\text{odd}} 1(t \equiv 1 \mod 2)$$

$$= \prod_{n \equiv t \mod 2} \mathbf{K}^{(n,n+1)}$$

NON-REVERSIBLE PT (OKABE ET AL, 2001)

- Deterministically alternate between **Odd** and **Even** swaps

$$\mathbf{K}_{\text{comm},t} = \mathbf{K}_{\text{even}} 1(t \equiv 0 \mod 2) + \mathbf{K}_{\text{odd}} 1(t \equiv 1 \mod 2)$$

$$= \prod_{n \equiv t \mod 2} \mathbf{K}^{(n,n+1)}$$

- Non-reversible PT trajectories have momentum!

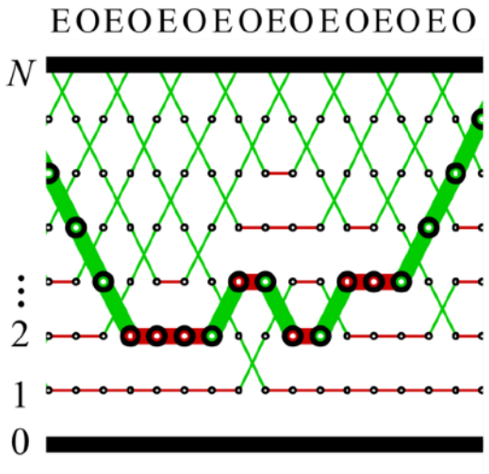
NON-REVERSIBLE PT (OKABE ET AL, 2001)

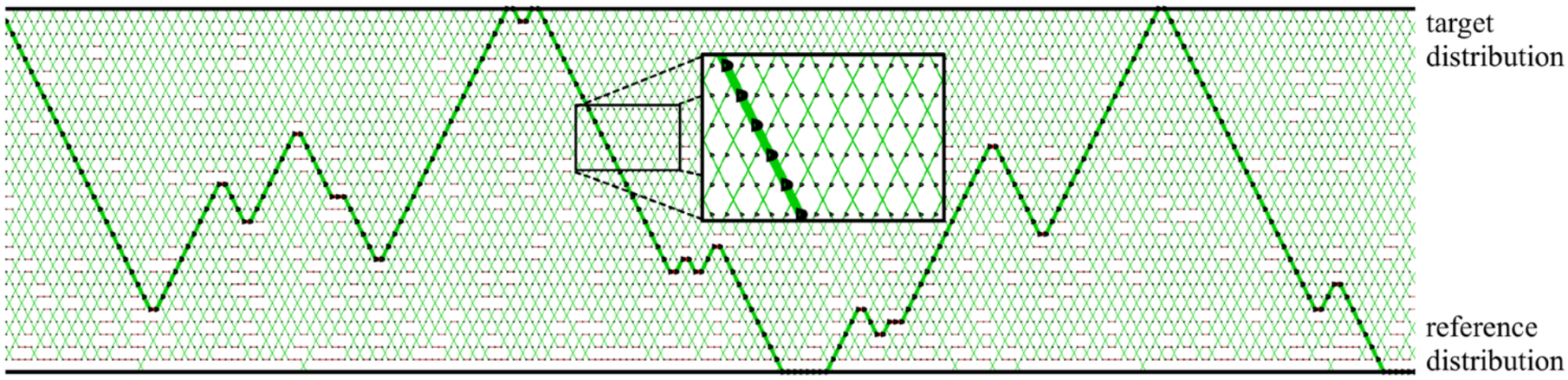
- Deterministically alternate between **Odd** and **Even** swaps

$$\mathbf{K}_{\text{comm},t} = \mathbf{K}_{\text{even}} 1(t \equiv 0 \mod 2) + \mathbf{K}_{\text{odd}} 1(t \equiv 1 \mod 2)$$

$$= \prod_{n \equiv t \mod 2} \mathbf{K}^{(n,n+1)}$$

- Non-reversible PT trajectories have momentum!





NON-REVERSIBLE PT (OKABE ET AL, 2001)

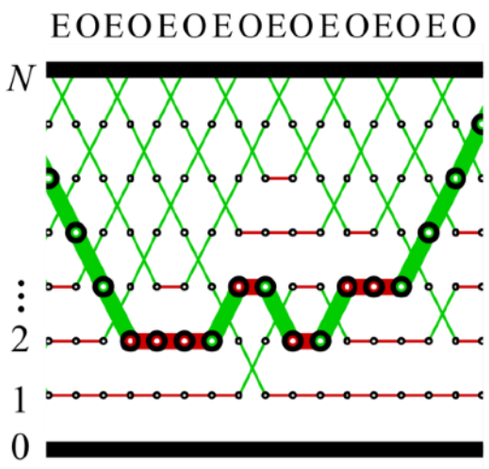
- Deterministically alternate between **Odd** and **Even** swaps

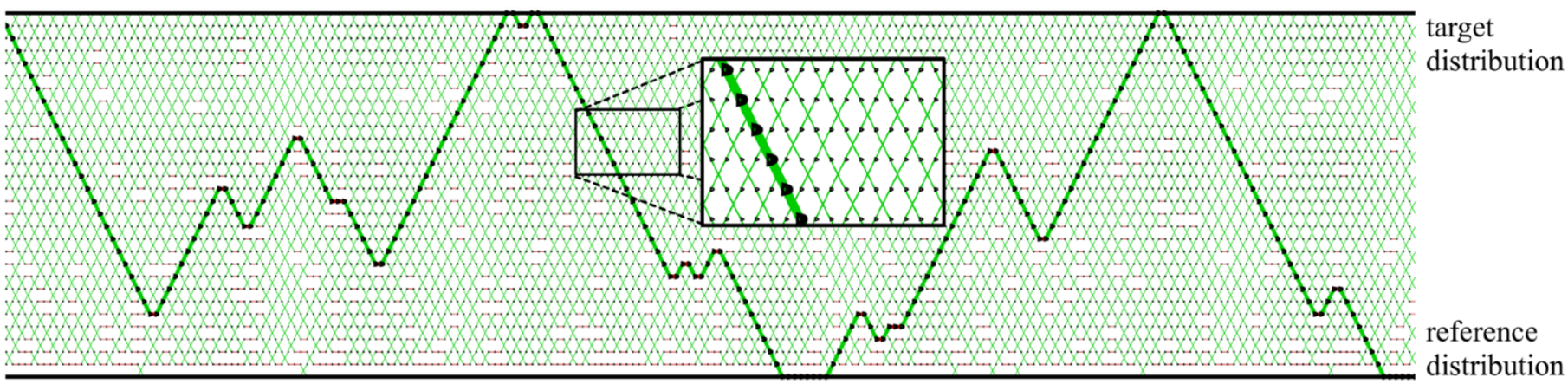
$$\mathbf{K}_{\text{comm},t} = \mathbf{K}_{\text{even}} 1(t \equiv 0 \mod 2) + \mathbf{K}_{\text{odd}} 1(t \equiv 1 \mod 2)$$

$$= \prod_{n \equiv t \mod 2} \mathbf{K}^{(n,n+1)}$$

- Non-reversible PT trajectories have momentum!
  - If swap for machine m is accepted

$$\epsilon_{t+1}^{m} = \epsilon_{t}^{m}$$





NON-REVERSIBLE PT (OKABE ET AL, 2001)

- Deterministically alternate between **Odd** and **Even** swaps

$$\mathbf{K}_{\text{comm},t} = \mathbf{K}_{\text{even}} 1(t \equiv 0 \mod 2) + \mathbf{K}_{\text{odd}} 1(t \equiv 1 \mod 2)$$

$$= \prod_{n \equiv t \mod 2} \mathbf{K}^{(n,n+1)}$$

- Non-reversible PT trajectories have momentum!
  - If swap for machine m is accepted

$$\epsilon_{t+1}^{m} = \epsilon_{t}^{m}$$

  - Otherwise

$$\epsilon_{t+1}^{m} = -\epsilon_{t}^{m}$$