An Estimation-Control Duality and its extension to unknown distributions

Self Study Seminar

Ranjan Sarkar
(24AI92R01)

Supervisor:
Dr. Prabhat Kumar Mishra

Department of Artificial Intelligence
IIT Kharagpur

Date - December 1, 2025

Objective

- To **understand** the research paper:

  S. Talebi, A. Taghvaei, and M. Mesbahi, "*Data-driven Optimal Filtering for Linear Systems with Unknown Noise Covariances*". NeurIPS, 2023.

- **Regenerate** the simulation results.

Motivating Example

Want to know my exact position?

- Can't see due to Heavy Fog
- GPS Sensor (noisy)
- How much noisy are the sensors !!

Want to **Estimate !**

Generated by Gemini Veo

Discrete Dynamical System

Consider the **Linear Time-invariant (LTI) dynamical system**,

State: $x_t \in \mathbb{R}^n$

System Dynamics

$$x_{t+1} = Ax_t + \xi_t$$

Observation: $y_t \in \mathbb{R}^m$

Observation Model

$$y_t = Hx_t + \omega_t$$

Discrete Dynamical System

Consider the **Linear Time-invariant (LTI) dynamical system**,

State: $x_t \in \mathbb{R}^n$

System Dynamics

$$x_{t+1} = Ax_t + \xi_t$$

Observation: $y_t \in \mathbb{R}^m$

Observation Model

$$y_t = Hx_t + \omega_t$$

- Noises are coming from unknown distribution with zero mean and given covariances.

$$\xi_t \sim (\mathbf{0}, \mathbf{Q}) \qquad \omega_t \sim (\mathbf{0}, \mathbf{R})$$

- Noises are uncorrelated with $\boldsymbol{x}_0$ and with each other.

State Estimation

Consider the **Linear Time-invariant (LTI) dynamical system**,

$$\begin{aligned} x_{t+1} &= Ax_t + \xi_t \\ y_t &= Hx_t + \omega_t \end{aligned}$$

State Estimation

Consider the **Linear Time-invariant (LTI) dynamical system**,

$$\begin{aligned} x_{t+1} &= Ax_t + \xi_t \\ y_t &= Hx_t + \omega_t \end{aligned}$$

Given, History of Observations: $\mathcal{Y}_t := \{y_{0:t}\} = \{y_0, y_1, \cdots, y_{t-1}\}$

State Estimation

Consider the **Linear Time-invariant (LTI) dynamical system**,

$$x_{t+1} = Ax_t + \xi_t$$
$$y_t = Hx_t + \omega_t$$

Given, History of Observations: $\mathcal{Y}_t := \{y_{0:t}\} = \{y_0, y_1, \cdots, y_{t-1}\}$

- **Minimum Mean Squared Error (MSE) Estimator:**

$$\hat{x}_t = \arg\min_{\hat{x} \in \mathcal{F}(\mathcal{Y}_t)} \mathbb{E}\|x_t - \hat{x}\|^2$$

- **Minimum MSE Linear Estimator:**

$$\hat{x}_t = \arg\min_{\hat{x} \in \mathcal{L}(\mathcal{Y}_t)} \mathbb{E}\|x_t - \hat{x}\|^2$$

Here, $\mathcal{F}(\mathcal{Y}_t)$: the space of all functions of the history of the observation signal $\mathcal{Y}_t$

$\mathcal{L}(\mathcal{Y}_t)$: the space of all **linear** functions of the history of the observation signal $\mathcal{Y}_t$

Kalman Filter (KF)

KF Recursive formula by considering prior estimates only:

$$\hat{x}_{t+1} = A\hat{x}_t + \overset{\substack{\text{Kalman Gain} \\ \uparrow}}{L_t} \underbrace{(y_t - H\hat{x}_t)}_{\text{Innovation Term}}$$

Kalman Filter (KF)

KF Recursive formula by considering prior estimates only:

Kalman Gain

$$\hat{x}_{t+1} = A\hat{x}_t + L_t \underbrace{(y_t - H\hat{x}_t)}_{\text{Innovation Term}}$$

Minimum MSE Linear Estimator:

How to get optimal L_t ?

$$\hat{x}_t = \arg\min_{\hat{x} \in \mathcal{L}(\mathcal{Y}_t)} \mathbb{E}\,\|x_t - \hat{x}\|^2$$

R. E. Kalman, ASME. Journal of Basic Engineering, 1960

Kalman Filter (KF)

Initialization: $\hat{x}_0, P_0$

Estimation Error Covariance at time t

$$\boxed{P_t = \mathbb{E}\left[(x_t - \hat{x}_t)(x_t - \hat{x}_t)^\intercal\right]}$$

* **Chapter 2** [F. Lewis. “Optimal Estimation with an Introduction to Stochastic Control Theory”, Wiley-Interscience, 1986]

Kalman Filter (KF)

Initialization: $\hat{x}_0, P_0$

Estimation Error Covariance at time t

$$\boxed{P_t = \mathbb{E}\left[(x_t - \hat{x}_t)(x_t - \hat{x}_t)^\intercal\right]}$$

*** A Priori KF Update Formula:**

Kalman Gain: $L_t := AP_tH^\intercal \left(HP_tH^\intercal + R\right)^{-1}$

State estimate (Prior): $\hat{x}_{t+1} = A\hat{x}_t + L_t\left(y_t - H\hat{x}_t\right)$

Error covariance (Prior): $P_{t+1} = AP_tA^\intercal + Q - AP_tH^\intercal\left(HP_tH^\intercal + R\right)^{-1} HP_tA^\intercal$

Depends on Q and R!

* **Chapter 2** [F. Lewis. "Optimal Estimation with an Introduction to Stochastic Control Theory", Wiley-Interscience, 1986]

Kalman Filter (KF)

Visualization

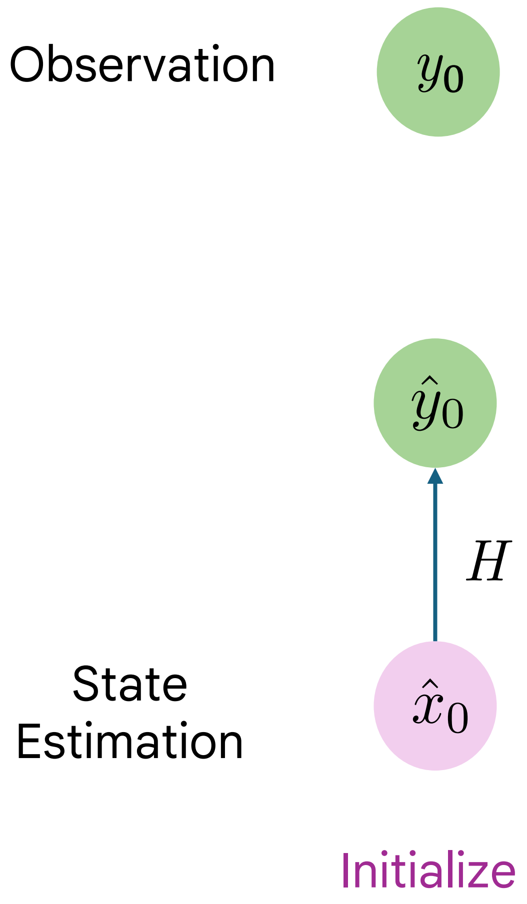
Observation y_0

State Estimation $\hat{x}_0$

Initialize

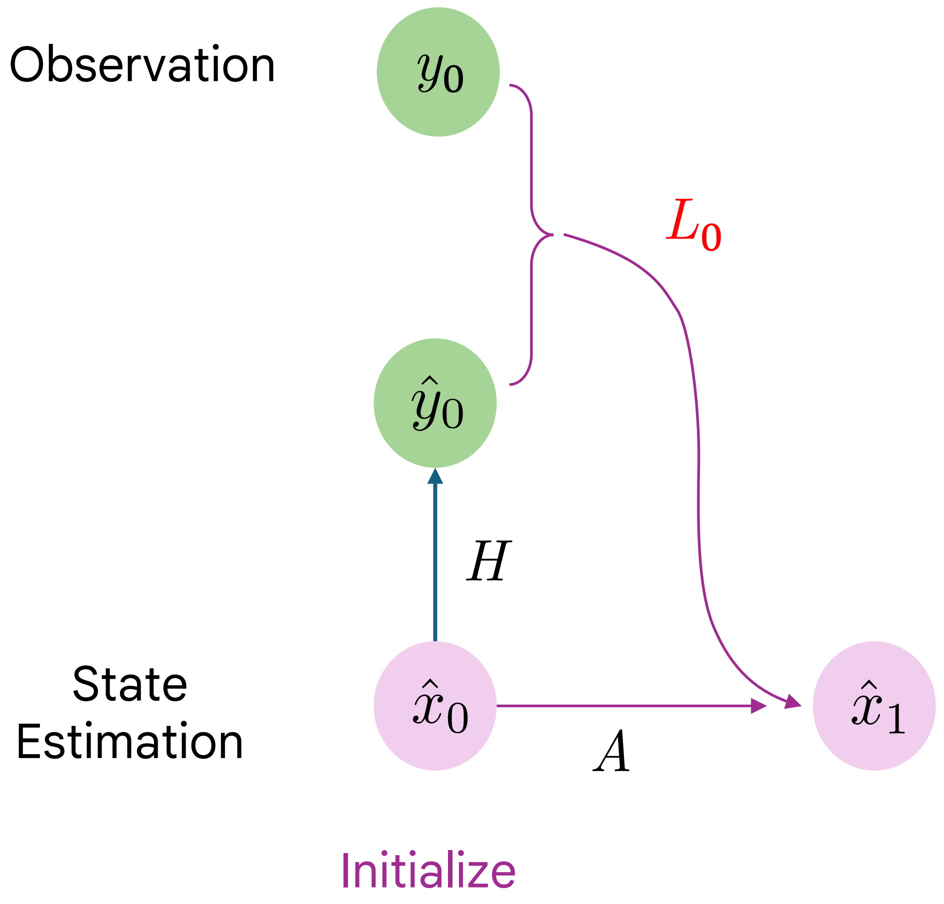
Kalman Filter (KF)

Visualization



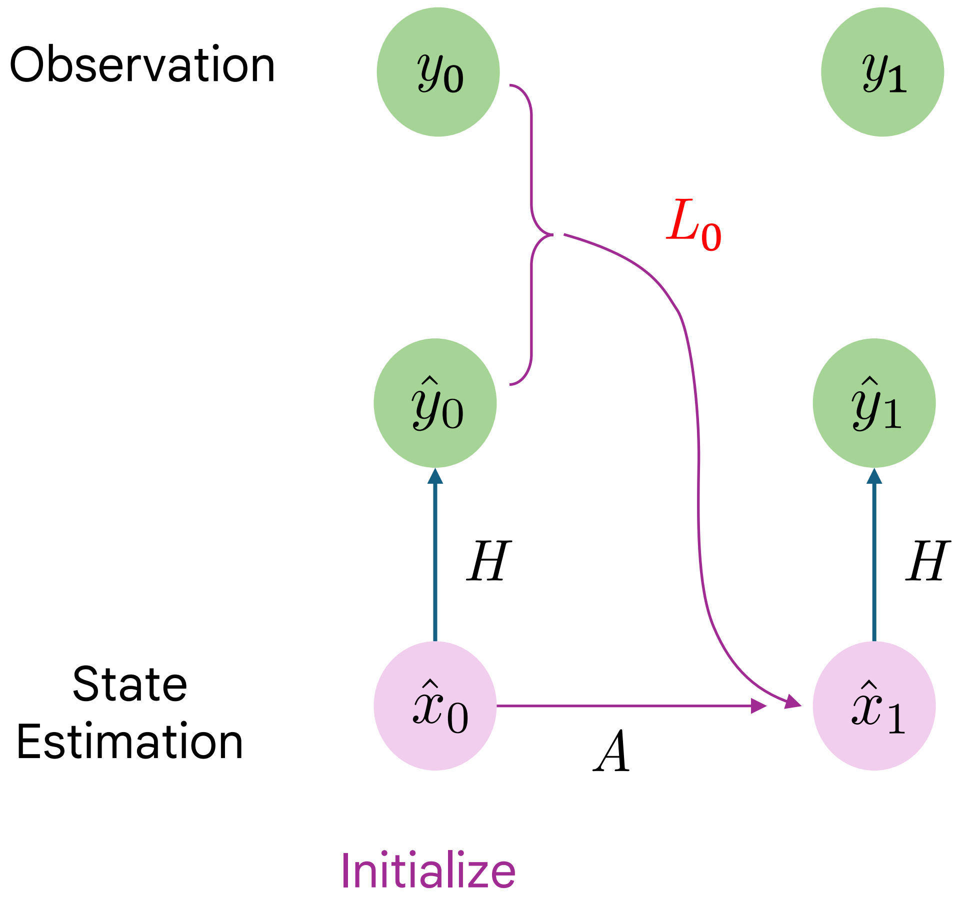
Kalman Filter (KF)

Visualization



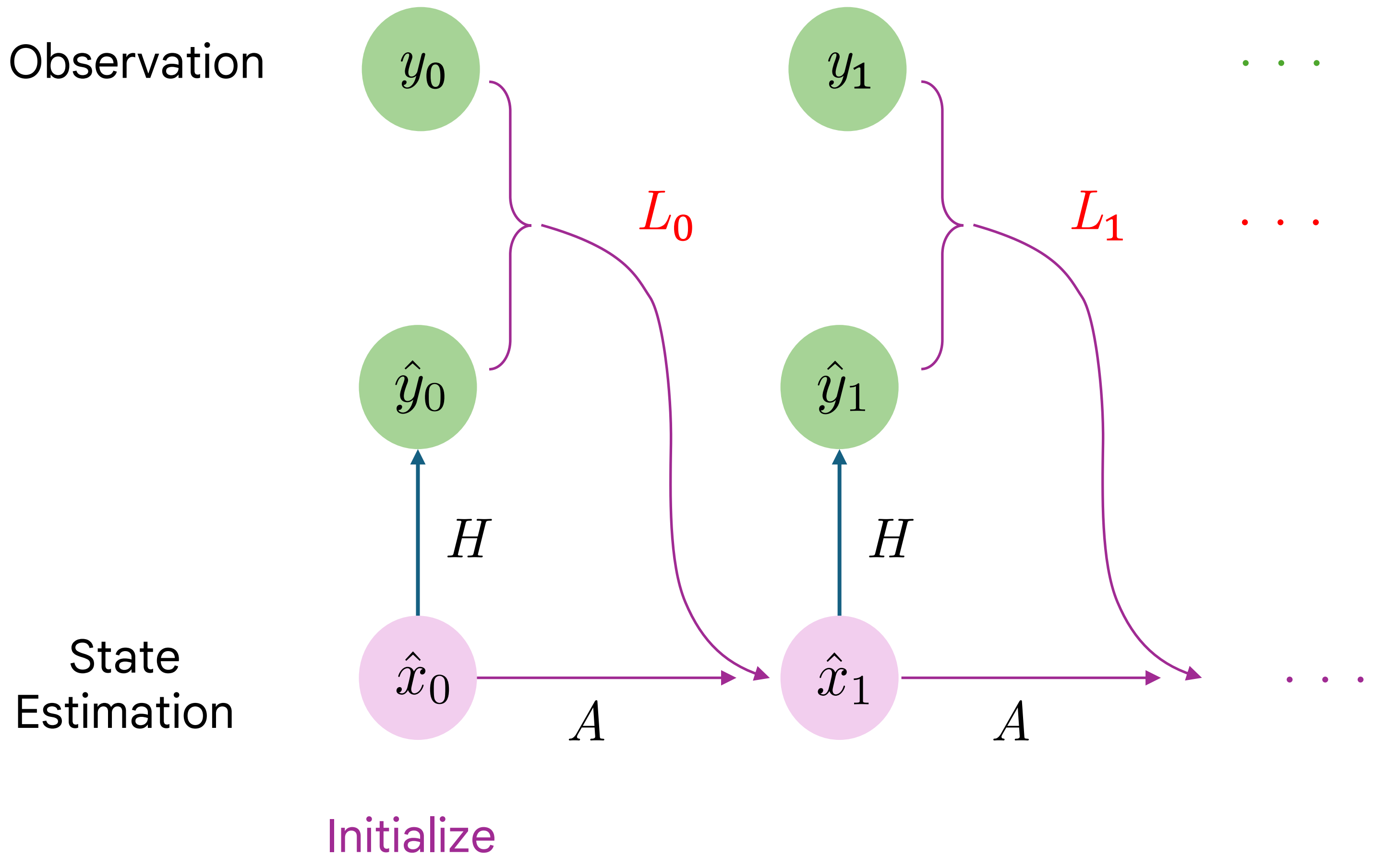
Kalman Filter (KF)

Visualization



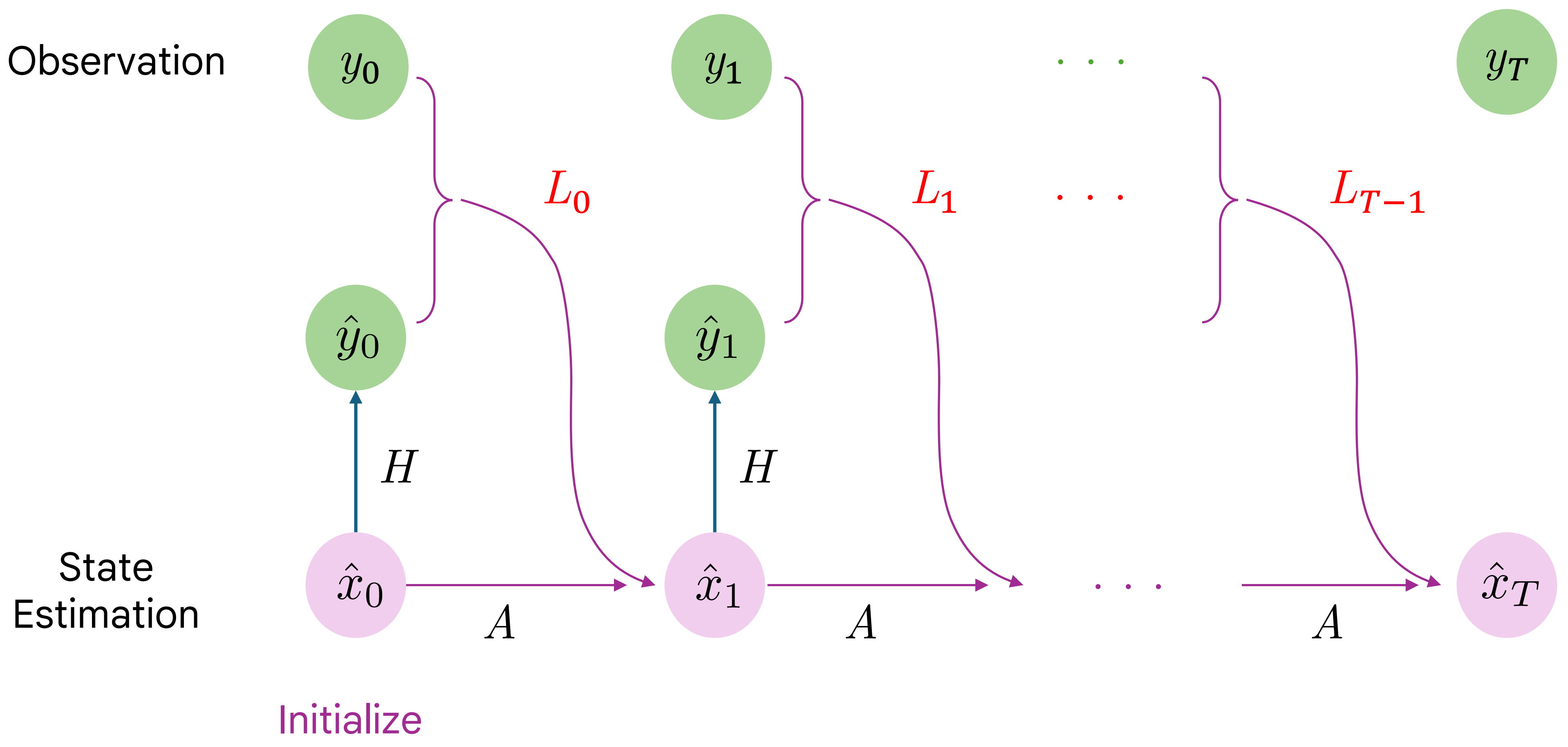
Kalman Filter (KF)

Visualization



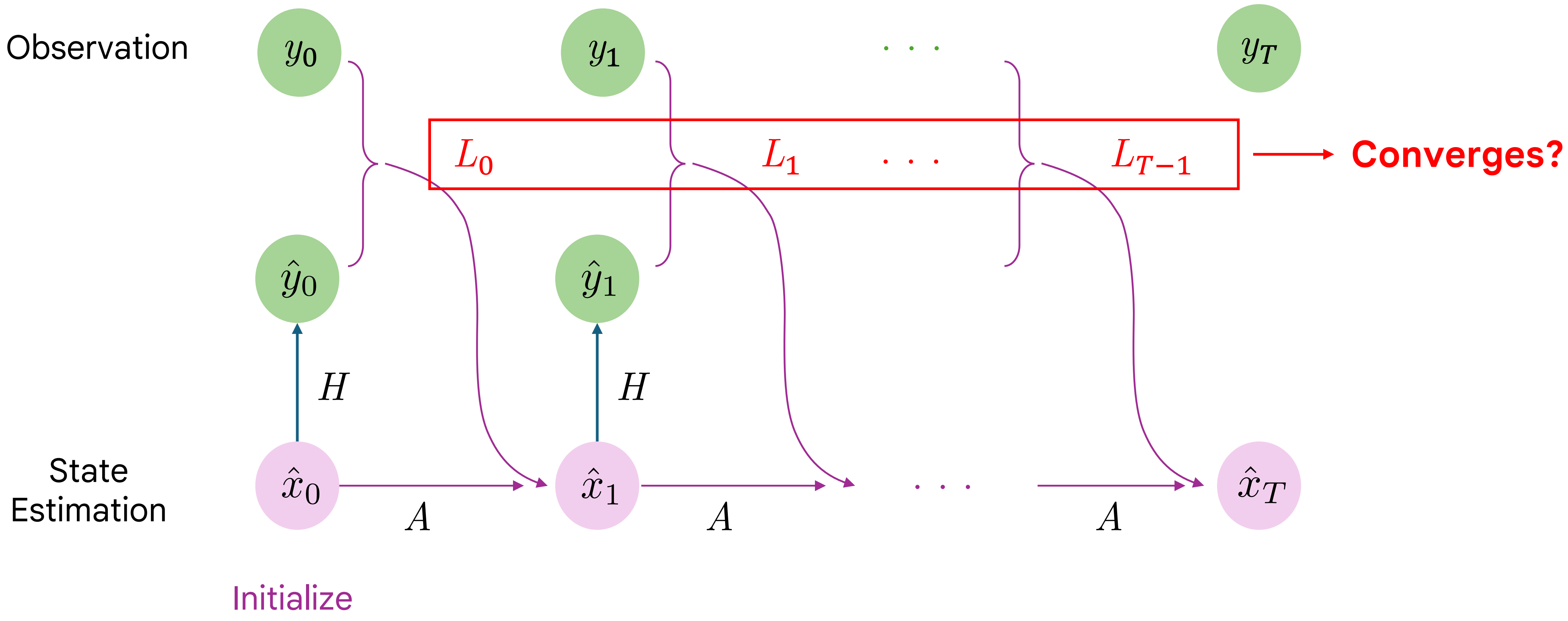
Kalman Filter (KF)

Visualization



Kalman Filter (KF)

Visualization



Steady-State Kalman Gain

Assumption

The pair (A, H) is **detectable**, and the pair $(A, \sqrt{Q})$ is **stabilizable**, where $\sqrt{Q}$ is the unique positive semidefinite square root of Q.

Steady-State Kalman Gain

Assumption

The pair (A, H) is **detectable**, and the pair $(A, \sqrt{Q})$ is **stabilizable**, where $\sqrt{Q}$ is the unique positive semidefinite square root of Q.

Under this assumption *
P_t will converge to **steady-state Covariance** P_∞.

* **Theorem 4.11** [H. Kwakernaak, R. Sivan. "Linear Optimal Control Systems". Wiley-interscience, 1969]

Steady-State Kalman Gain

Assumption

The pair (A, H) is **detectable**, and the pair $(A, \sqrt{Q})$ is **stabilizable**, where $\sqrt{Q}$ is the unique positive semidefinite square root of Q.

Under this assumption *
P_t will converge to **steady-state Covariance** P_∞.

Steady-state Kalman Gain

$$\boxed{L_\infty = A P_\infty H^\top \left(H P_\infty H^\top + R\right)^{-1}}$$

* **Theorem 4.11** [H. Kwakernaak, R. Sivan. "Linear Optimal Control Systems". Wiley-interscience, 1969]

Steady-State Kalman Gain

Assumption

The pair (A, H) is **detectable**, and the pair $(A, \sqrt{Q})$ is **stabilizable**, where $\sqrt{Q}$ is the unique positive semidefinite square root of Q.

Under this assumption *
P_t will converge to **steady-state Covariance** P_∞.

Steady-state Kalman Gain

$$\boxed{L_\infty = A P_\infty H^\top \left(H P_\infty H^\top + R \right)^{-1}}$$

Depends on Q and R!

* **Theorem 4.11** [H. Kwakernaak, R. Sivan. "Linear Optimal Control Systems". Wiley-interscience, 1969]

Learning Problem *

Given	Not Given
• **Known** Dynamics (A, H are **known**). • We have the history of observations (noisy): $\mathcal{Y}_t = \{y_0, y_1, \cdots, y_{t-1}\}$	• **Unknown** noise covariances (Q, R). • We don't have the ground-truth measurement of state (x_t).

* S. Talebi et al., "Data-driven Optimal Filtering for Linear Systems with Unknown Noise Covariances". NeurIPS '23.

Learning Problem *

Given

- **Known** Dynamics (A, H are **known**).
- We have the history of observations (noisy):

$$\mathcal{Y}_t = \{y_0, y_1, \cdots, y_{t-1}\}$$

Not Given

- **Unknown** noise covariances (Q, R).
- We don't have the ground-truth measurement of state (x_t).

Learn	Steady-State Kalman Gain (L_∞)

* S. Talebi et al., "Data-driven Optimal Filtering for Linear Systems with Unknown Noise Covariances". NeurIPS '23.

Learn L_∞

- Kalman Filter: $\hat{x}_{t+1} = A\hat{x}_t + L_t(y_t - H\hat{x}_t)$

Learn L_∞

- Kalman Filter: $\hat{x}_{t+1} = A\hat{x}_t + L_t(y_t - H\hat{x}_t)$

- Considering constant gain ($\boldsymbol{L}$) in Kalman filter update:

$$\hat{x}_{t+1}(L) = A\hat{x}_t + L(y_t - H\hat{x}_t)$$

Learn L_∞

- Kalman Filter: $\hat{x}_{t+1} = A\hat{x}_t + L_t(y_t - H\hat{x}_t)$

- Considering constant gain ($\boldsymbol{L}$) in Kalman filter update:

$$\hat{x}_{t+1}(L) = A\hat{x}_t + L(y_t - H\hat{x}_t) \quad \Rightarrow \quad \hat{x}_{t+1}(L) = \underbrace{(A - LH)}_{A_L}\hat{x}_t + Ly_t$$

Learn L_∞

- Kalman Filter: $\hat{x}_{t+1} = A\hat{x}_t + L_t(y_t - H\hat{x}_t)$

- Considering constant gain ($\boldsymbol{L}$) in Kalman filter update:

$$\hat{x}_{t+1}(L) = A\hat{x}_t + L(y_t - H\hat{x}_t) \quad \Rightarrow \quad \hat{x}_{t+1}(L) = \underbrace{(A - LH)}_{A_L}\hat{x}_t + Ly_t$$

- Expanding this for $\boldsymbol{t = 0}$ to $\boldsymbol{T-1}$, we get:

$$\hat{x}_T(L) = A_L^T\hat{x}_0 + \sum_{t=0}^{T-1} A_L^{T-t-1}Ly_t$$

Get state estimate at time $\boldsymbol{T}$ using $\hat{\boldsymbol{x}}_0$

Learn L_∞

- Kalman Filter: $\hat{x}_{t+1} = A\hat{x}_t + L_t(y_t - H\hat{x}_t)$

- Considering constant gain (L) in Kalman filter update:

$$\hat{x}_{t+1}(L) = A\hat{x}_t + L(y_t - H\hat{x}_t) \quad \Rightarrow \quad \hat{x}_{t+1}(L) = \underbrace{(A - LH)}_{A_L}\hat{x}_t + Ly_t$$

- Expanding this for $t = 0$ to $T{-}1$, we get:

$$\hat{x}_T(L) = A_L^T\hat{x}_0 + \sum_{t=0}^{T-1} A_L^{T-t-1}Ly_t$$

Get state estimate at time T using $\hat{x}_0$

MSE: $\min_L \quad \mathbb{E}\left\|x_T - \hat{x}_T(L)\right\|^2$

x_t 's are not accessible !

Estimation-Control Duality

System Dynamics (Forward)

$$
\begin{aligned}
x_{t+1} &= Ax_t + \xi_t \\
y_t &= Hx_t + \omega_t
\end{aligned}
$$

Estimation-Control Duality

System Dynamics (Forward)

$$x_{t+1} = Ax_t + \xi_t$$
$$y_t = Hx_t + \omega_t$$

Dual Dynamics (Backward)

$$z_t = A^\intercal z_{t+1} - H^\intercal u_{t+1}$$

Given, $z_T = a \in \mathbb{R}^n$

* **Appendix C** [S. Talebi et al., NeurIPS '23]

Estimation-Control Duality

System Dynamics (Forward)

$$x_{t+1} = Ax_t + \xi_t$$
$$y_t = Hx_t + \omega_t$$

MSE Cost

$$\mathbb{E}\,\|x_T - \hat{x}_T(L)\|^2$$

Dual Dynamics (Backward)

$$z_t = A^\intercal z_{t+1} - H^\intercal u_{t+1}$$

Given, $z_T = a \in \mathbb{R}^n$

Estimation-Control Duality

System Dynamics (Forward)

$$x_{t+1} = Ax_t + \xi_t$$
$$y_t = Hx_t + \omega_t$$

MSE Cost

$$\mathbb{E}\,\|x_T - \hat{x}_T(L)\|^2$$

Dual Dynamics (Backward)

$$z_t = A^\intercal z_{t+1} - H^\intercal u_{t+1}$$

Given, $z_T = a \in \mathbb{R}^n$

LQR Cost (Finite Horizon)

$$J_T^{\mathrm{LQR}}\left(a, u_{1:T+1}\right) = z_0^\intercal P_0 z_0 + \sum_{t=1}^{T} z_t^\intercal Q z_t + u_t^\intercal R u_t$$

Estimation-Control Duality

System Dynamics (Forward)

$$x_{t+1} = Ax_t + \xi_t$$
$$y_t = Hx_t + \omega_t$$

MSE Cost

$$\mathbb{E}\left\|x_T - \hat{x}_T(L)\right\|^2$$

Dual Dynamics (Backward)

$$z_t = A^\intercal z_{t+1} - H^\intercal u_{t+1}$$

Given, $z_T = a \in \mathbb{R}^n$

LQR Cost (Finite Horizon)

$$J_T^{\mathrm{LQR}}\left(a, u_{1:T+1}\right) = z_0^\intercal P_0 z_0 + \sum_{t=1}^{T} z_t^\intercal Q z_t + u_t^\intercal R u_t$$

Duality*

$$\mathbb{E}\left[\left|a^\intercal x_T - a^\intercal \hat{x}_T(L)\right|^2\right] = J_T^{\mathrm{LQR}}\left(a, u_{1:T+1} \middle| u_i = L^\intercal z_i\right)$$

* **Appendix C** [S. Talebi et al., NeurIPS '23]

Estimation-Control Duality

System Dynamics (Forward)

$$x_{t+1} = Ax_t + \xi_t$$
$$y_t = Hx_t + \omega_t$$

MSE Cost

$$\mathbb{E}\,\|x_T - \hat{x}_T(L)\|^2$$

Dual Dynamics (Backward)

$$z_t = A^\intercal z_{t+1} - H^\intercal u_{t+1}$$

Given, $z_T = a \in \mathbb{R}^n$

LQR Cost (Finite Horizon)

$$J_T^{\mathrm{LQR}}\Big(a, u_{1:T+1}\Big) = z_0^\intercal P_0 z_0 + \sum_{t=1}^{T} z_t^\intercal Q z_t + u_t^\intercal R u_t$$

Duality*

Estimation $$\mathbb{E}\left[\left|a^\intercal x_T - a^\intercal \hat{x}_T(L)\right|^2\right] = J_T^{\mathrm{LQR}}\Big(a, u_{1:T+1} \big| u_i = L^\intercal z_i\Big)$$ Control

Stochastic — Deterministic

* **Appendix C** [S. Talebi et al., NeurIPS '23]

Learn L_∞

- In particular, choose $\boldsymbol{a} = \boldsymbol{H_i}$ ⟵ $\boldsymbol{H_i}^\mathsf{T}$ is the $\boldsymbol{i}$-th row of the matrix $\boldsymbol{H}$.

* **Proposition 1** [S. Talebi et al., NeurIPS '23]

Learn L_∞

- In particular, choose $\boldsymbol{a} = \boldsymbol{H_i}$ ⟵ $\boldsymbol{H_i}^\mathsf{T}$ is the $\boldsymbol{i}$-th row of the matrix $\boldsymbol{H}$.

- By Estimation-Control Duality *,

$$\mathbb{E}\left\|y_T - H\hat{x}_T(L)\right\|^2 = \sum_{i=1}^{m} J_T^{\mathrm{LQR}}\left(H_i, u_{1:T+1} | u_i = L^\mathsf{T} z_i\right) + \mathrm{tr}\left[R\right]$$

* **Proposition 1** [S. Talebi et al., NeurIPS '23]

Learn L_∞

- In particular, choose $\boldsymbol{a} = \boldsymbol{H_i}$ ⟵ $\boldsymbol{H_i}^\intercal$ is the $\boldsymbol{i}$-th row of the matrix $\boldsymbol{H}$.

- By Estimation-Control Duality *,

$$\mathbb{E}\left\|y_T - H\hat{x}_T(L)\right\|^2 = \sum_{i=1}^{m} J_T^{\mathrm{LQR}}\Big(H_i, u_{1:T+1} | u_i = L^\intercal z_i\Big) + \mathrm{tr}\left[R\right]$$

Surrogate Objective (Finite time horizon)

$$J_T^{\mathrm{est}}(L) := \mathbb{E}\left\|y_T - \hat{y}_T(L)\right\|^2$$

$$\hat{y}_T(L) := H\hat{x}_T(L)$$

* **Proposition 1** [S. Talebi et al., NeurIPS '23]

Learn L_∞

Dual Dynamics (Backward)

$$z_t = A^\intercal z_{t+1} - H^\intercal u_{t+1} \qquad \text{Given, } z_T = a \in \mathbb{R}^n$$

LQR Cost (Finite Horizon)

$$J_T^{\mathrm{LQR}}\left(a, u_{1:T+1}\right) = z_0^\intercal P_0 z_0 + \sum_{t=1}^{T} z_t^\intercal Q z_t + u_t^\intercal R u_t$$

Substituting the dual dynamics:

$$\min_{L} \; J_T^{\mathrm{est}}(L) = \mathrm{tr}\left[X_T(L) H^\intercal H\right] + \mathrm{tr}[R]$$

where, $$X_T(L) := A_L^T P_0 \left(A_L^\intercal\right)^T + \sum_{t=0}^{T-1} A_L^t \left(Q + L R L^\intercal\right) \left(A_L^\intercal\right)^t$$

Learn L_∞

- **Surrogate Objective (Finite time horizon):**

$$\min_{L} \; J_T^{\text{est}}(L) = \operatorname{tr}\left[X_T(L) H^\intercal H\right] + \operatorname{tr}[R]$$

where, $$X_T(L) := A_L^T P_0 \left(A_L^\intercal\right)^T + \sum_{t=0}^{T-1} A_L^t \left(Q + LRL^\intercal\right) \left(A_L^\intercal\right)^t$$

Learn L_∞

- **Surrogate Objective (Finite time horizon)**:

$$\min_{L} \; J_T^{\text{est}}(L) = \operatorname{tr}\left[X_T(L) H^\intercal H\right] + \operatorname{tr}[R]$$

where, $$X_T(L) := A_L^T P_0 \left(A_L^\intercal\right)^T + \sum_{t=0}^{T-1} A_L^t \left(Q + LRL^\intercal\right)\left(A_L^\intercal\right)^t$$

- When Spectral radius (Largest absolute eigen value) $\rho(A_L) < 1$,

  $X_T(L)$ will converge to $X_{(L)}$ as $T \to \infty$.

Learn L_∞

- **Surrogate Objective (Finite time horizon):**

$$\min_{L} \; J_T^{\text{est}}(L) = \operatorname{tr}\left[X_T(L) H^\intercal H\right] + \operatorname{tr}[R]$$

where, $$X_T(L) := A_L^T P_0 \left(A_L^\intercal\right)^T + \sum_{t=0}^{T-1} A_L^t \left(Q + LRL^\intercal\right)\left(A_L^\intercal\right)^t$$

- When Spectral radius (Largest absolute eigen value) $\rho(A_L) < 1$,

  $X_T(L)$ will converge to $X_{(L)}$ as $T \to \infty$.

- Set of Schur stabilizing gains, $\mathcal{S} := \{L \in \mathbb{R}^{n \times m} : \rho(A_L) < 1\}$

Optimization in Steady-State

Surrogate Objective (in Steady-state)

$$\lim_{T\to\infty} J_T^{\text{est}}(L) = \operatorname{tr}\left[X_{(L)} H^\intercal H\right] + \operatorname{tr}\left[R\right]$$

Optimization in Steady-State

Surrogate Objective (in Steady-state)

$$\lim_{T\to\infty} J_T^{\text{est}}(L) = \operatorname{tr}\left[X_{(L)}H^\intercal H\right] + \operatorname{tr}\left[R\right]$$

Constrained Optimization Problem in steady-state

$$\begin{aligned}
\min_{L\in\mathcal{S}} \quad & J(L) := \operatorname{tr}\left[X_{(L)}H^\intercal H\right] \\
\text{s.t.} \quad & X_{(L)} = A_L X_{(L)} A_L^\intercal + Q + LRL^\intercal \\
\text{where,} \quad & \mathcal{S} := \{L \in \mathbb{R}^{n\times m} : \rho(A_L) < 1\}
\end{aligned}$$

Optimization in Steady-State

Surrogate Objective (in Steady-state)

$$\lim_{T\to\infty} J_T^{\text{est}}(L) = \operatorname{tr}\left[X_{(L)} H^\intercal H\right] + \operatorname{tr}\left[R\right]$$

Constrained Optimization Problem in steady-state

$$\begin{aligned} \min_{L\in\mathcal{S}} \quad & J(L) := \operatorname{tr}\left[X_{(L)} H^\intercal H\right] \\ \text{s.t.} \quad & X_{(L)} = A_L X_{(L)} A_L^\intercal + Q + L R L^\intercal \\ \text{where,} \quad & \mathcal{S} := \{L \in \mathbb{R}^{n\times m} : \rho(A_L) < 1\} \end{aligned}$$

$\boldsymbol{Q},\ \boldsymbol{R}$

are unknown !

Learn L_∞

Optimization in Steady-State

Surrogate Objective (in Steady-state)

$$\lim_{T\to\infty} J_T^{\text{est}}(L) = \operatorname{tr}\left[X_{(L)} H^\intercal H\right] + \operatorname{tr}\left[R\right]$$

Constrained Optimization Problem in steady-state

$$\begin{aligned} &\min_{L\in\mathcal{S}} \quad J(L) := \operatorname{tr}\left[X_{(L)} H^\intercal H\right] \\ &\text{s.t.} \quad X_{(L)} = A_L X_{(L)} A_L^\intercal + Q + LRL^\intercal \\ &\text{where,} \quad \mathcal{S} := \{L \in \mathbb{R}^{n\times m} : \rho(A_L) < 1\} \end{aligned}$$

$Q,\ R$
are unknown !

Standard Kalman Filter & Duality will not work

Data driven approach

Learn L_∞

Data-driven approach

Estimate the **Objective**

$$J(L)$$

Learn L_∞

Data-driven approach

Estimate the **Objective**

$$J(L) \longleftarrow \text{Steady-state Cost}$$

Learn L_∞

Data-driven approach

Estimate the **Objective**

$J(L)$ ⟵ Steady-state Cost

We have finite horizon data.

Learn L_∞

Data-driven approach

Estimate the **Objective**

$$J(L) \longleftarrow$$ Steady-state Cost

We have finite horizon data.

Estimate $\underbrace{\textbf{truncated objective}}$

$$J_T(L) = \mathbb{E}\,\|y_T - \hat{y}_T(L)\|^2$$

Learn L_∞

Data-driven approach

Estimate the **Objective**

$J(L) \longleftarrow$ Steady-state Cost

We have finite horizon data.

$\downarrow$

Estimate **truncated objective**

$$\underbrace{J_T(L) \;=\; \mathbb{E}\,\|y_T - \hat{y}_T(L)\|^2}_{} \longleftarrow \widehat{J}_T(L) = \frac{1}{M}\sum_{i=1}^{M} \overbrace{\|y_T^{(i)} - \hat{y}_T^{(i)}(L)\|^2}^{\mathrm{SE}\left(L, \mathcal{Y}_T^{(i)}\right)}$$

(Squared Error)

Learn L_∞

Data-driven approach

Estimate the **Objective**

$J(L)$ $\longleftarrow$ Steady-state Cost

We have finite horizon data.

$\downarrow$

Estimate **truncated objective**

$$J_T(L) \;=\; \mathbb{E}\,\|y_T - \hat{y}_T(L)\|^2 \;\longleftarrow\; \widehat{J}_T(L) = \frac{1}{M}\sum_{i=1}^{M} \overbrace{\|y_T^{(i)} - \hat{y}_T^{(i)}(L)\|^2}^{\mathrm{SE}\left(L,\, \mathcal{Y}_T^{(i)}\right)}$$

(Squared Error)

Estimate **gradient** of **truncated objective**

$$\nabla J_T(L)$$

Learn L_∞

Data-driven approach

Estimate the **Objective**

$J(L)$ ← Steady-state Cost

We have finite horizon data.

Estimate **truncated objective**

$$J_T(L) = \mathbb{E}\,\|y_T - \hat{y}_T(L)\|^2 \longleftarrow \widehat{J}_T(L) = \frac{1}{M}\sum_{i=1}^{M} \overbrace{\|y_T^{(i)} - \hat{y}_T^{(i)}(L)\|^2}^{\mathrm{SE}\left(L, \mathcal{Y}_T^{(i)}\right)}$$

(Squared Error)

Estimate **gradient** of **truncated objective**

$$\nabla J_T(L) \longleftarrow \nabla \widehat{J}_T(L) = \frac{1}{M}\sum_{i=1}^{M} \nabla\,\mathrm{SE}\left(L, \mathcal{Y}_T^{(i)}\right)$$

Learn L_∞

Role of Duality in estimating gradient

Estimate the **gradient** of **truncated objective**

$$\nabla \widehat{J}_T(L) = \frac{1}{M}\sum_{i=1}^{M} \nabla \, \mathrm{SE}\left(L, \mathcal{Y}_T^{(i)}\right)$$

$$= \frac{1}{M}\sum_{i=1}^{M} \nabla \left\| \underbrace{y_T^{(i)} - \hat{y}_T^{(i)}(L)}_{e_T^{(i)}(L)} \right\|^2$$

Learn L_∞

Role of Duality in estimating gradient

Estimate the **gradient** of **truncated objective**

$$\nabla \widehat{J}_T(L) = \frac{1}{M}\sum_{i=1}^{M} \nabla\, \mathrm{SE}\left(L, \mathcal{Y}_T^{(i)}\right)$$

$$= \frac{1}{M}\sum_{i=1}^{M} \nabla \Big\| \underbrace{y_T^{(i)} - \hat{y}_T^{(i)}(L)}_{e_T^{(i)}(L)} \Big\|^2$$

$$= -\frac{2}{M}\sum_{i=1}^{M}\sum_{t=0}^{T-1} \underbrace{(A_L^\intercal)^{T-t-1} H^\intercal e_T^{(i)}}_{z_{t+1}(L)}(L)\, e_t^{(i)}(L)^\intercal$$

Note: $\nabla \widehat{J}_T(L)$ does not depend on $\boldsymbol{Q}$, $\boldsymbol{R}$.

[**Lemma 3,** S. Talebi et al,. NeurIPS, 2023]

$\longleftarrow$ Adjoint state from Dual Dynamics

Algorithm

Batch Gradient Descent

Require:

$\boldsymbol{A},\ \boldsymbol{H},\ \hat{x}_0,\ \boldsymbol{P}_0.$

Hyperparameters:

$\boldsymbol{T}$: Trajectory Length, $\boldsymbol{M}$: Batch size,

η : Step Length, $\boldsymbol{k}$: No. of iterations.

Algorithm

Batch Gradient Descent

Require:

$\boldsymbol{A},\ \boldsymbol{H},\ \hat{x}_0,\ \boldsymbol{P}_0.$

Hyperparameters:

$\boldsymbol{T}$: Trajectory Length, $\boldsymbol{M}$: Batch size,

η : Step Length, $\boldsymbol{k}$: No. of iterations.

Initialize

$\boldsymbol{L} = \boldsymbol{L}_0$

Algorithm

Batch Gradient Descent

Require:

$\boldsymbol{A},\ \boldsymbol{H},\ \hat{x}_0,\ \boldsymbol{P}_0.$

Hyperparameters:

$\boldsymbol{T}$: Trajectory Length, $\boldsymbol{M}$: Batch size,

η : Step Length, $\boldsymbol{k}$: No. of iterations.

Initialize
$\boldsymbol{L} = \boldsymbol{L}_0$

Sample **M** trajectories.
$\{\mathcal{Y}_T^{(i)}\}_{i=1}^{M}$

Algorithm

Require:

$\boldsymbol{A},\ \boldsymbol{H},\ \hat{\boldsymbol{x}}_0,\ \boldsymbol{P}_0.$

Hyperparameters:

$\boldsymbol{T}$: Trajectory Length, $\boldsymbol{M}$: Batch size,

η : Step Length, $\boldsymbol{k}$: No. of iterations.

Algorithm

Batch Gradient Descent

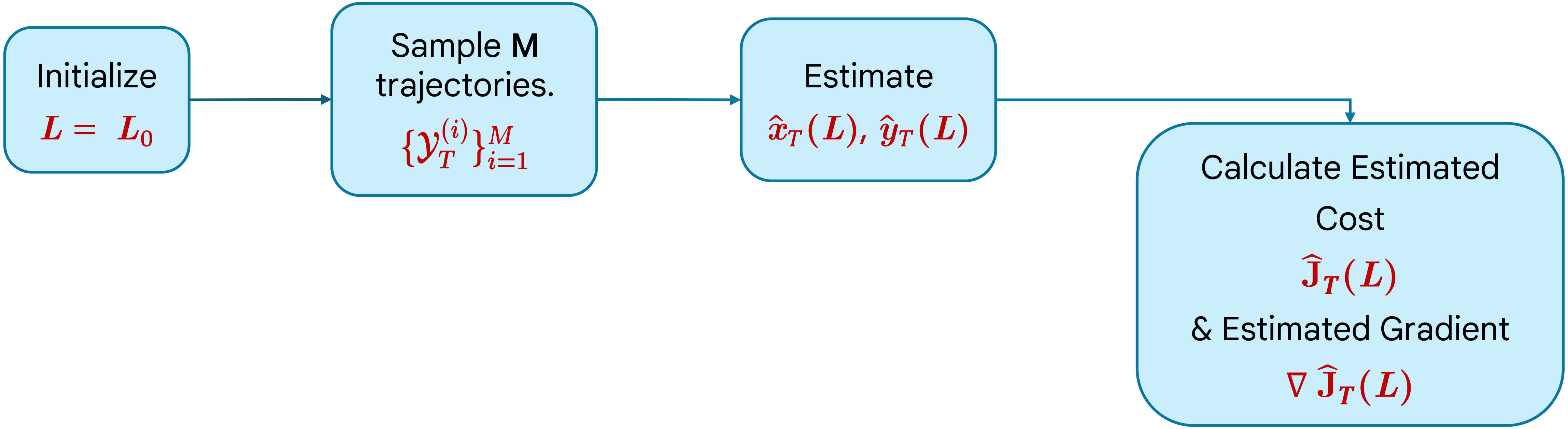
Require:

$\boldsymbol{A},\ \boldsymbol{H},\ \hat{x}_0,\ \boldsymbol{P}_0.$

Hyperparameters:

$\boldsymbol{T}$: Trajectory Length, $\boldsymbol{M}$: Batch size,

η : Step Length, $\boldsymbol{k}$: No. of iterations.



Algorithm

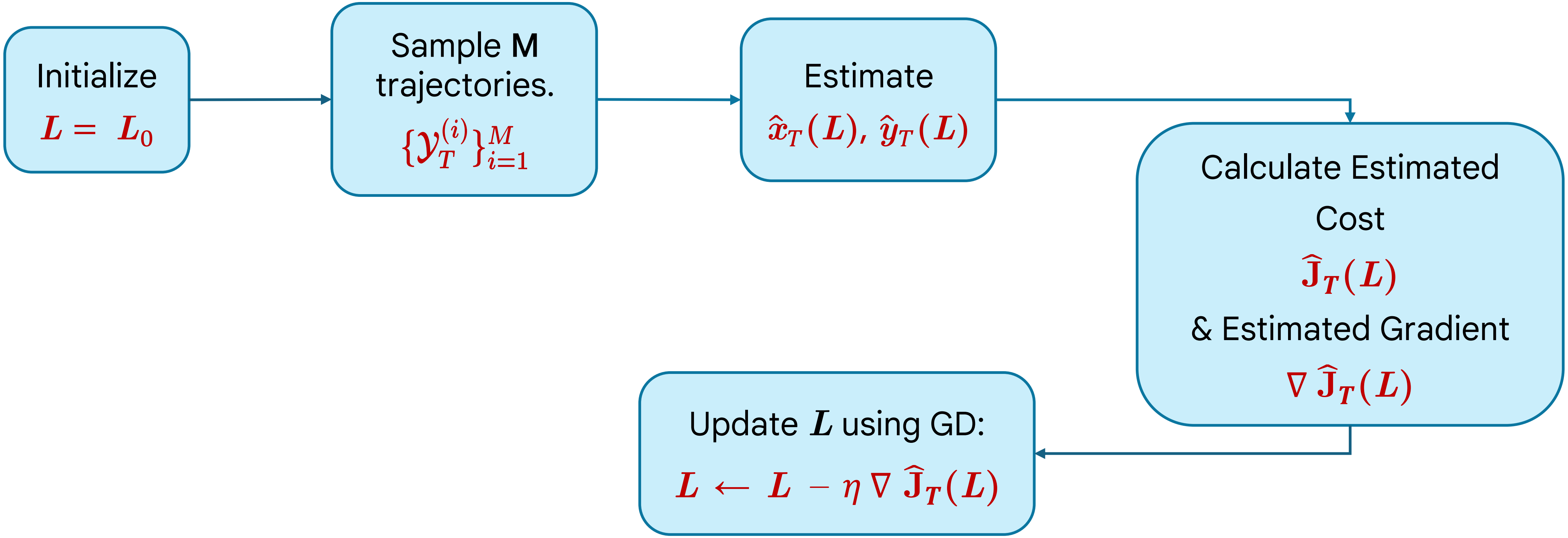
Batch Gradient Descent

Require:

$\boldsymbol{A},\ \boldsymbol{H},\ \hat{x}_0,\ \boldsymbol{P}_0.$

Hyperparameters:

$\boldsymbol{T}$: Trajectory Length, $\boldsymbol{M}$: Batch size,
η : Step Length, $\boldsymbol{k}$: No. of iterations.



Algorithm

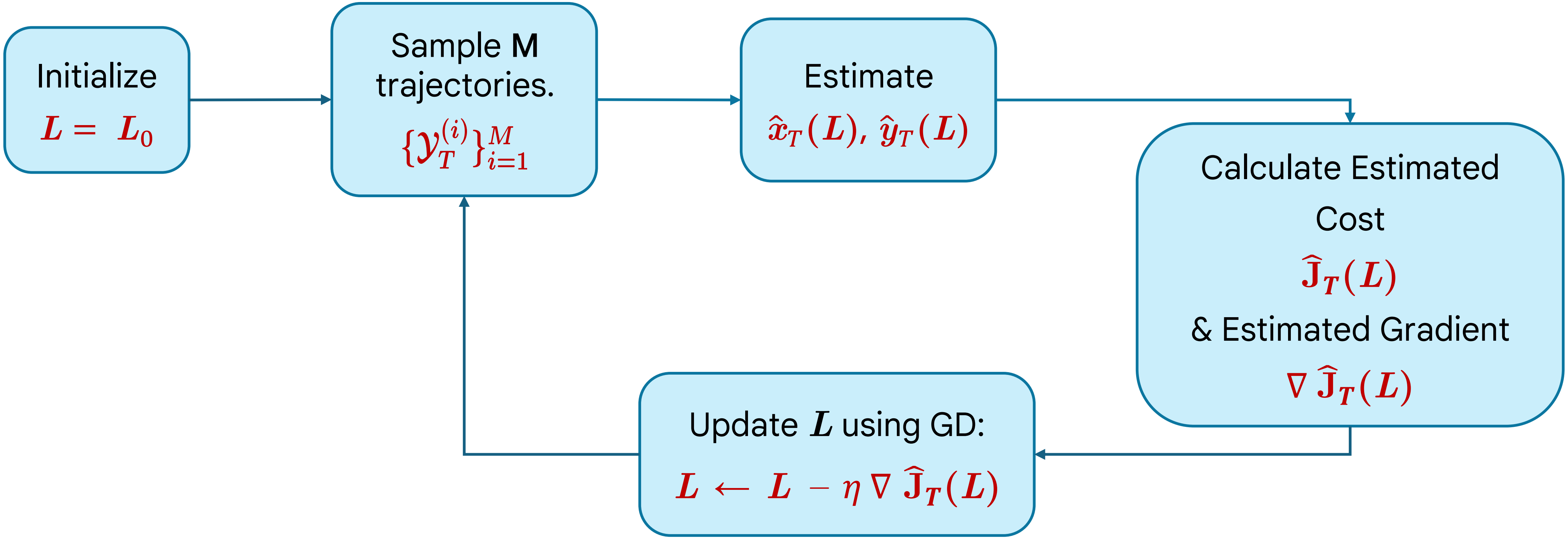
Batch Gradient Descent

Require:

$\boldsymbol{A}$, $\boldsymbol{H}$, $\hat{x}_0$, $\boldsymbol{P}_0$.

Hyperparameters:

$\boldsymbol{T}$: Trajectory Length, $\boldsymbol{M}$: Batch size,
η : Step Length, $\boldsymbol{k}$: No. of iterations.



Algorithm

Batch Gradient Descent

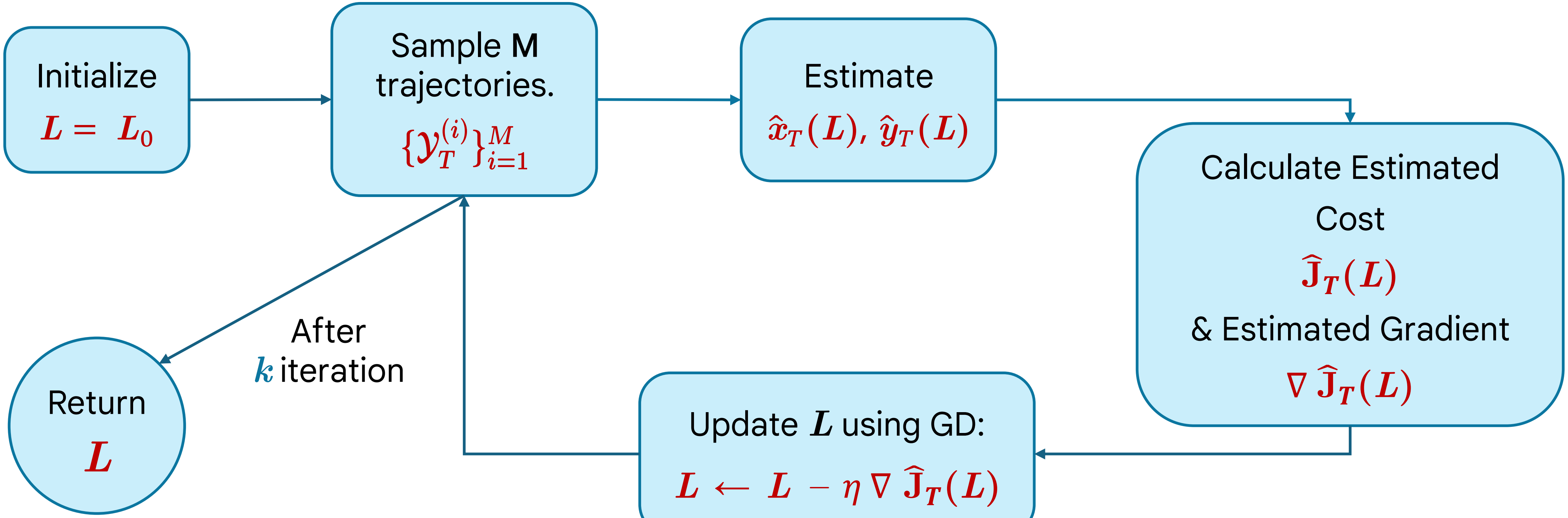
Require:

$\boldsymbol{A},\ \boldsymbol{H},\ \hat{\boldsymbol{x}}_0,\ \boldsymbol{P}_0.$

Hyperparameters:

$\boldsymbol{T}$: Trajectory Length, $\boldsymbol{M}$: Batch size,

η : Step Length, $\boldsymbol{k}$: No. of iterations.



Research Questions

- How to initialize $\boldsymbol{L_0}$?
- How to choose $\boldsymbol{T}$ (Trajectory length), $\boldsymbol{M}$ (Batch-size) ?
- How the choices of $\boldsymbol{L_0}$, $\boldsymbol{T}$, $\boldsymbol{M}$ will affect the convergence rate?

Convergence Analysis

We are getting the sequence of gain: $\boldsymbol{L_0}, \boldsymbol{L_1}, \ldots, \boldsymbol{L_k}$

Convergence Analysis

We are getting the sequence of gain: $\boldsymbol{L_0,\ L_1,\ \dots\ ,\ L_k}$

- Satisfy the **Stability Constraint** i.e. $L \in S$ **?**

Convergence Analysis

We are getting the sequence of gain: $\boldsymbol{L_0},\ \boldsymbol{L_1},\ \ldots\ ,\boldsymbol{L_k}$

- Satisfy the **Stability Constraint** i.e. $L \in \mathcal{S}$ **?**

  Define Sublevel Set for some $\boldsymbol{\alpha} > 0,\ \boldsymbol{\mathcal{S}}_{\alpha} := \{\boldsymbol{L} \in \mathbb{R}^{n\times m} : \boldsymbol{J}(\boldsymbol{L}) \leq \alpha\}$

Convergence Analysis

We are getting the sequence of gain: $\boldsymbol{L_0},\ \boldsymbol{L_1},\ \ldots\ ,\ \boldsymbol{L_k}$

- Satisfy the **Stability Constraint** i.e. $L \in \mathcal{S}$ **?**

  Define Sublevel Set for some $\boldsymbol{\alpha} > 0,\ \boldsymbol{\mathcal{S}}_\alpha := \{\boldsymbol{L} \in \mathbb{R}^{\boldsymbol{n} \times \boldsymbol{m}} : \boldsymbol{J}(\boldsymbol{L}) \leq \alpha\}$

- How to ensure **Cost Decay** in each step of GD **?**

Convergence Analysis

$$\|\nabla \widehat{J}_T(L) - \nabla J(L)\|$$

Convergence Analysis

Error Bounds

$$\|\nabla \widehat{J}_T(L) - \nabla J(L)\| \;\leq \|\nabla \widehat{J}_T(L) - \nabla J_T(L)\| + \|\nabla J_T(L) - \nabla J(L)\|$$

Convergence Analysis

Error Bounds

$$\|\nabla \widehat{J}_T(L) - \nabla J(L)\| \;\leq\; \underbrace{\|\nabla \widehat{J}_T(L) - \nabla J_T(L)\|}_{\text{Concentration Error}} + \underbrace{\|\nabla J_T(L) - \nabla J(L)\|}_{\text{Truncation Error}}$$

Convergence Analysis

$$\|\nabla \widehat{J}_T(L) - \nabla J(L)\| \le \underbrace{\|\nabla \widehat{J}_T(L) - \nabla J_T(L)\|}_{\text{Concentration Error}} + \underbrace{\|\nabla J_T(L) - \nabla J(L)\|}_{\text{Truncation Error}}$$

- **Concentration Error Bound** [1]

$$\mathbb{P}\Big[\,\Big\|\nabla \widehat{J}_T(L) - \nabla J_T(L)\Big\| \ge s\Big] \le 2n\exp\Big[-M\,c(s,L)\Big]$$

Depends on **M, L**

[1] **Proposition 4,** [2] **Proposition 5** [S. Talebi et al,. NeurIPS '23]

Convergence Analysis

Error Bounds

$$\|\nabla \widehat{J}_T(L) - \nabla J(L)\| \leq \underbrace{\|\nabla \widehat{J}_T(L) - \nabla J_T(L)\|}_{\text{Concentration Error}} + \underbrace{\|\nabla J_T(L) - \nabla J(L)\|}_{\text{Truncation Error}}$$

- **Concentration Error Bound** [1]

$$\mathbb{P}\left[\left\|\nabla \widehat{J}_T(L) - \nabla J_T(L)\right\| \geq s\right] \leq 2n \exp\left[-M\, c(s, L)\right]$$

← Depends on **M, L**

- **Truncation Error Bound** [2]

$$\|\nabla J(L) - \nabla J_T(L)\| \leq \bar{\gamma}_L \sqrt{\rho(A_L)}^{T+1}$$

← Depends on **T, L**

[1] **Proposition 4,** [2] **Proposition 5** [S. Talebi et al,. NeurIPS '23]

Convergence Analysis

$$\|\nabla \widehat{J}_T(L) - \nabla J(L)\| \leq \underbrace{\|\nabla \widehat{J}_T(L) - \nabla J_T(L)\|}_{\text{Concentration Error}} + \underbrace{\|\nabla J_T(L) - \nabla J(L)\|}_{\text{Truncation Error}}$$

Concentration Error: Decays as **M** grows.

Truncation Error: Decays as **T** grows.

- **Concentration Error Bound** [1]

$$\mathbb{P}\left[\left\|\nabla \widehat{J}_T(L) - \nabla J_T(L)\right\| \geq s\right] \leq 2n \exp\left[-M\, c(s, L)\right]$$

← Depends on **M, L**

- **Truncation Error Bound** [2]

$$\|\nabla J(L) - \nabla J_T(L)\| \leq \bar{\gamma}_L \sqrt{\rho(A_L)}^{T+1}$$

← Depends on **T, L**

[1] **Proposition 4,** [2] **Proposition 5** [S. Talebi et al,. NeurIPS '23]

Convergence Analysis

Given **T** length **M** trajectories with **M** $> l(\delta)$, $\delta > 0$

Convergence Analysis

Biased Gradient

Given **T** length **M** trajectories with **M** $> l(\delta)$, $\delta > 0$

Suppose we have access to a biased estimate of the gradient $\nabla \widehat{J}(L)$

The following holds with probability $> 1 - \delta$

Convergence Analysis

Given **T** length **M** trajectories with **M** $> l(\delta)$, $\delta > 0$

Suppose we have access to a biased estimate of the gradient $\nabla \widehat{J}(L)$

The following holds with probability $> 1 - \delta$

There exists constants $s, s_0 > 0$ implying

$$\|\nabla \widehat{J}(L) - \nabla J(L)\|_F \leq s \|\nabla J(L)\|_F + s_0 \quad \text{for all } L \in S_\alpha \setminus \mathcal{C}_\tau$$

for some $\alpha > 0$.

τ-neighborhood of L^* (Optimal Gain)

Convergence Analysis

Linear rate of Convergence

Now, we have biased Gradient

Convergence Analysis

Linear rate of Convergence

Now, we have biased Gradient

GD algorithm starting from any $L_0 \in \mathcal{S}_\alpha \setminus \mathcal{C}_\tau$ with fixed step-size $\eta(\alpha, \gamma)$

Convergence Analysis

Linear rate of Convergence

Now, we have biased Gradient

GD algorithm starting from any $L_0 \in \mathcal{S}_\alpha \setminus \mathcal{C}_\tau$ with fixed step-size $\eta(\alpha, \gamma)$

- Then it generates a sequence of policies $\{\boldsymbol{L_k}\}$ that are stable (i.e. each $L_k \in S_\alpha$)

Satisfying the Constraint ✓

Convergence Analysis

Linear rate of Convergence

Now, we have biased Gradient

GD algorithm starting from any $L_0 \in \mathcal{S}_\alpha \setminus \mathcal{C}_\tau$ with fixed step-size $\eta(\alpha, \gamma)$

- Then it generates a sequence of policies $\{\boldsymbol{L_k}\}$ that are stable (i.e. each $L_k \in S_\alpha$) **Satisfying the Constraint ✓**

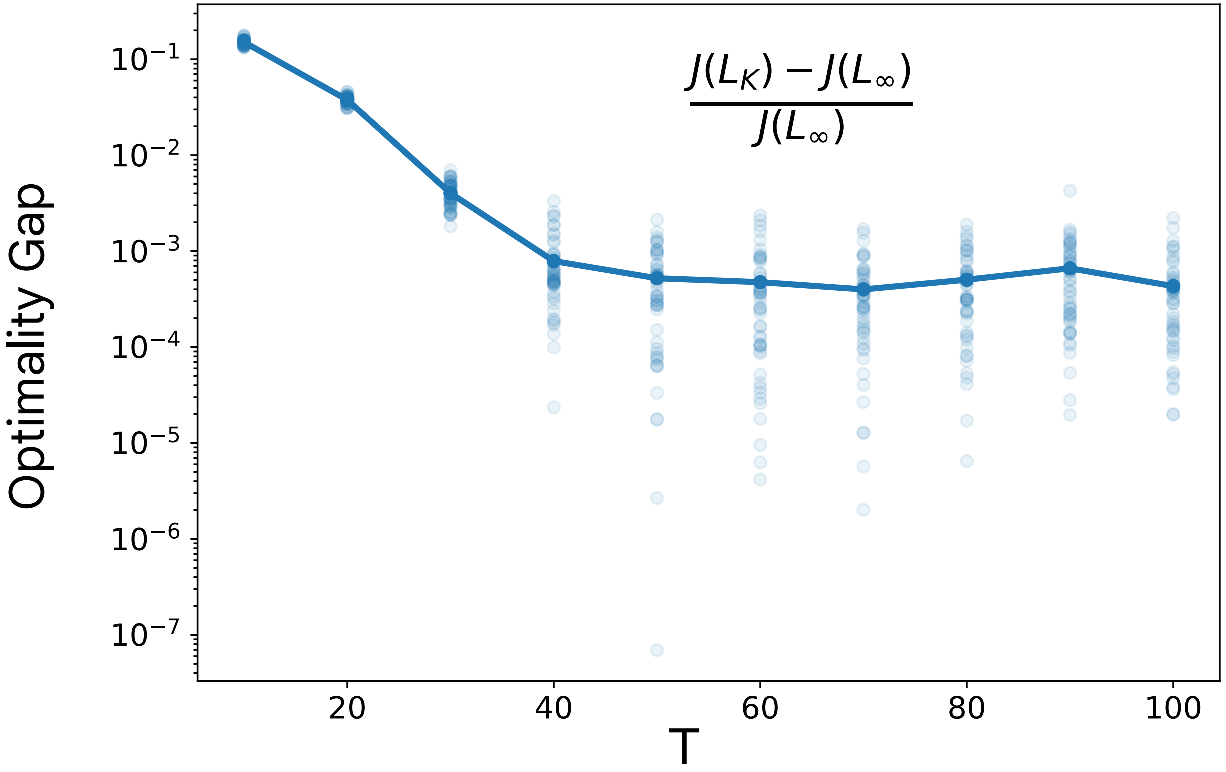
- **Decay** in cost value with **Linear convergence rate** before entering $\mathcal{C}_\tau$

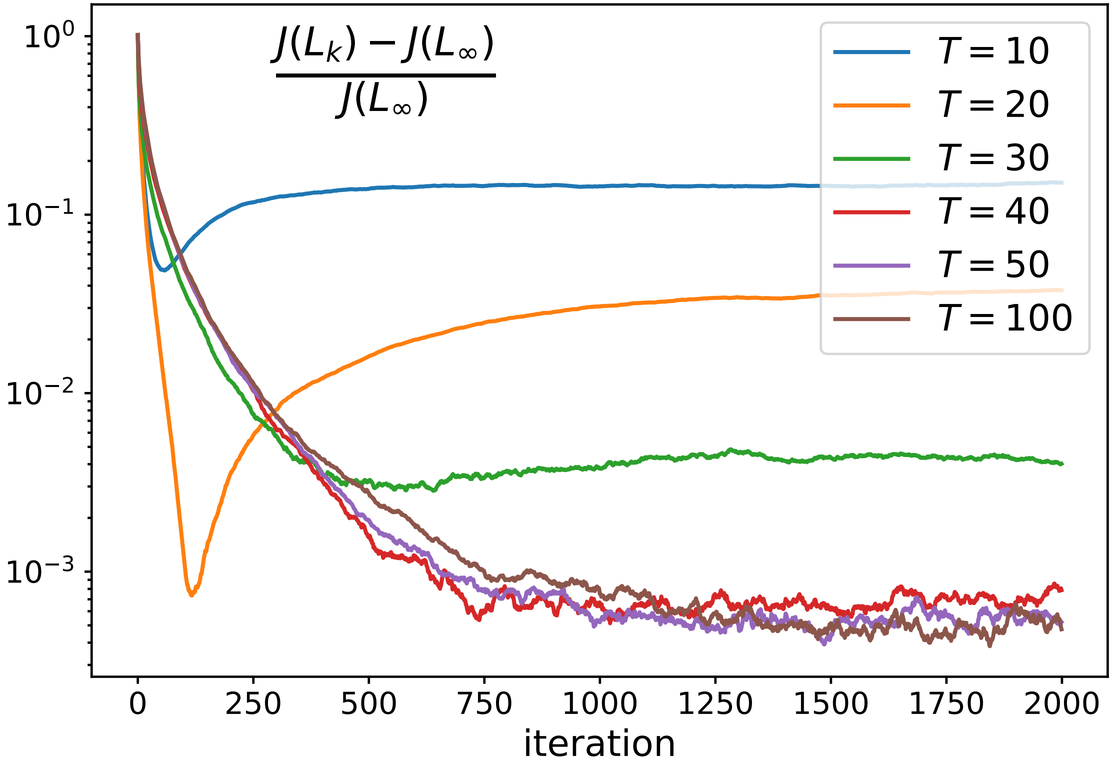
$$J(L_{k+1}) - J(L^*) \leq c_1(\alpha, \gamma, \eta)\left[J(L_k) - J(L^*)\right]$$

Theorem 1 [S. Talebi et al,. NeurIPS ‘23]

Simulation Result

Trajectory Length (T)



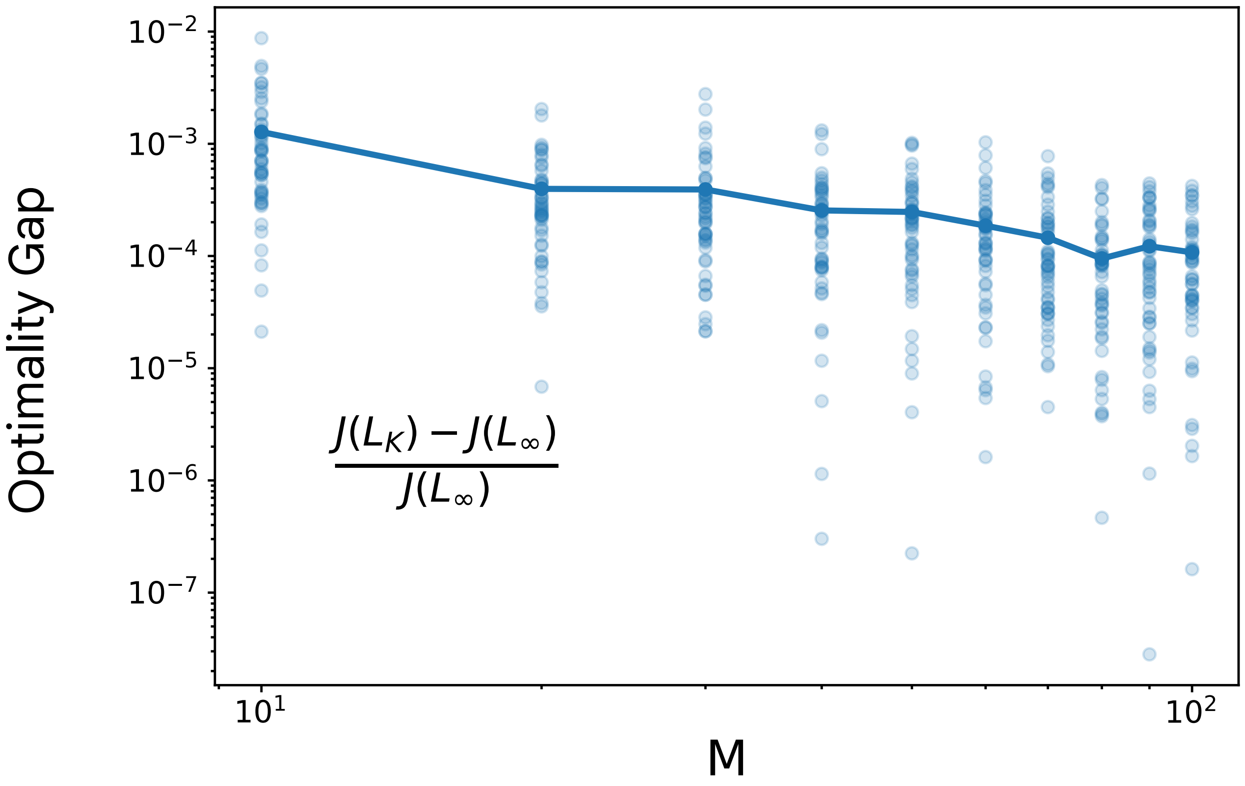


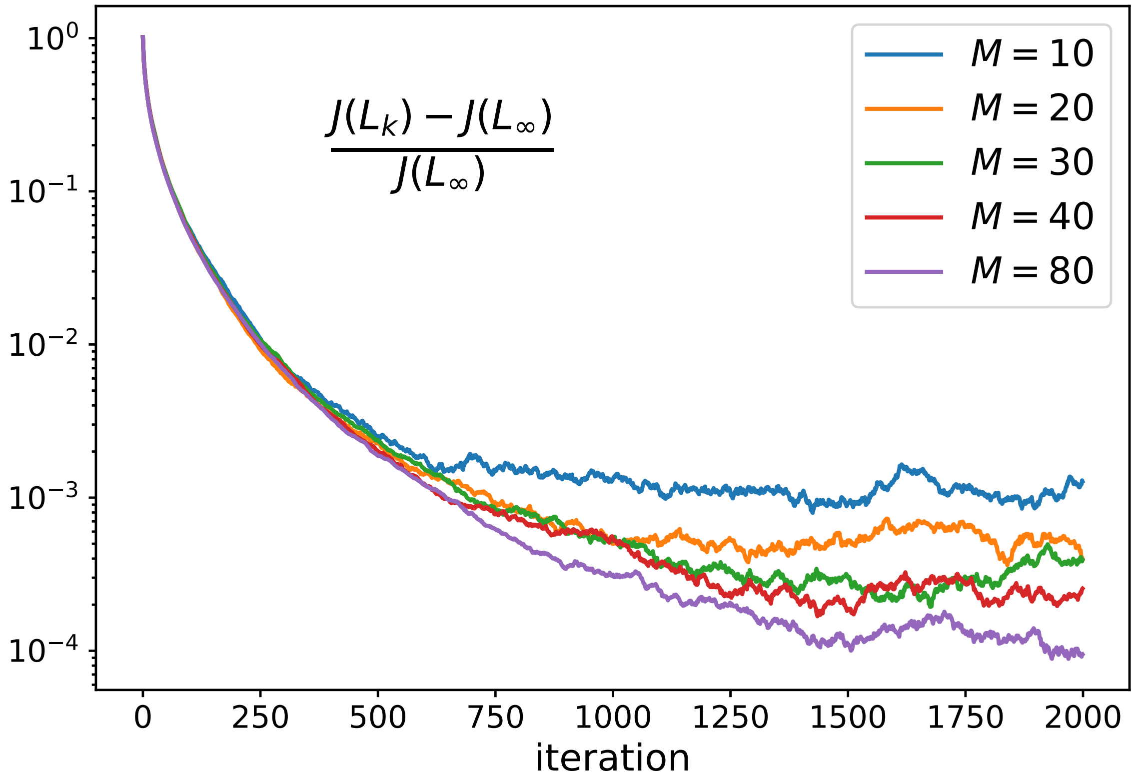
It shows **linear decay of optimality gap** w.r.t. **T**.

* Regenerated using code provided by authors.

Simulation Result

Batch-size (M)





It shows **linear decay of optimality gap** when the amount of data increases.

* Regenerated using code provided by authors.

Convergence Analysis

Consider **observable** $(\boldsymbol{A}, \boldsymbol{H})$, **bounded noise** $\boldsymbol{\xi_t}, \boldsymbol{\omega_t}$,

with (Stability Constraint) $\boldsymbol{L_0} \in \boldsymbol{\mathcal{S}}$ and step-size $\bar{\eta} := \dfrac{2}{9\,\ell(J(L_0))}$

Convergence Analysis

Consider **observable** $(\boldsymbol{A}, \boldsymbol{H})$, **bounded noise** $\boldsymbol{\xi_t}, \boldsymbol{\omega_t}$,

with (Stability Constraint) $\boldsymbol{L_0} \in \boldsymbol{\mathcal{S}}$ and step-size $\bar{\eta} := \dfrac{2}{9\,\ell(J(L_0))}$

For all $\epsilon > 0$, if

$$T \geq O(\ln(\frac{1}{\epsilon})), \quad M \geq O\Big(\frac{1}{\epsilon}\ln(\frac{1}{\delta})\ln(\ln(\frac{1}{\epsilon}))\Big) \quad \text{and} \quad k \geq O(\ln(\frac{1}{\epsilon}))$$

Trajectory Length — Batch-size — Iteration no.

Convergence Analysis

Consider **observable** $(\boldsymbol{A}, \boldsymbol{H})$, **bounded noise** $\boldsymbol{\xi_t}, \boldsymbol{\omega_t}$,

with (Stability Constraint) $\boldsymbol{L_0} \in \boldsymbol{\mathcal{S}}$ and step-size $\bar{\eta} := \dfrac{2}{9\,\ell(J(L_0))}$

For all $\epsilon > 0$, if

$$T \geq O(\ln(\frac{1}{\epsilon})), \quad M \geq O\Big(\frac{1}{\epsilon}\ln(\frac{1}{\delta})\ln(\ln(\frac{1}{\epsilon}))\Big) \quad \text{and} \quad k \geq O(\ln(\frac{1}{\epsilon}))$$

Trajectory Length — Batch-size — Iteration no.

Then Batch GD converges to ϵ-optimal gain i.e. $J(L_k) - J(L^*) \leq \epsilon$

with higher probability $(> 1 - \delta)$

Theorem 3 [S. Talebi et al,. NeurIPS '23]

Future Work

Current Algorithm

- Considers **prior** estimation.
- **Offline** Learning.

Plan

- Develop **Posterior** estimation.
- Develop **Online** Learning.
- Try RL algorithms when model $(\boldsymbol{A}, \boldsymbol{H})$ is unknown.

Thank You