Non-linear MPC

Robert Platt
Northeastern University

NonLinear Model Predictive Control

Given:

System: $x_{t+1} = Ax_t + Bu_t$

Cost function: $J(X,U) = x_T^T Q_T x_T + \sum_{t=1}^{T-1} x_t^T Q x_t + u_t^T R u_t$

where: $X = (x_1, \ldots, x_T)$

$U = (u_1, \ldots, u_{T-1})$

Initial state: x_1

Calculate: *U* that minimizes J(*X*,*U*)

NonLinear Model Predictive Control

Given:

System: $$\xcancel{x_{t+1} = Ax_t + Bu_t}$$

Cost function: $$J(X,U) = x_T^T Q_T x_T + \sum_{t=1}^{T-1} x_t^T Q x_t + u_t^T R u_t$$

where: $$X = (x_1, \ldots, x_T)$$ $$U = (u_1, \ldots, u_{T-1})$$

Initial state: x_1

Calculate: *U* that minimizes J(*X*,*U*)

NonLinear Model Predictive Control

Given:

System: $x_{t+1} = f(x_t, u_t)$

Cost function: $J(X,U) = x_T^T Q_T x_T + \sum_{t=1}^{T-1} x_t^T Q x_t + u_t^T R u_t$

where: $X = (x_1, \ldots, x_T)$

$U = (u_1, \ldots, u_{T-1})$

Initial state: x_1

Calculate: *U* that minimizes J(*X*,*U*)

NonLinear Model Predictive Control

Minimize: $$J(X,U) = x_T^T Q_T x_T + \sum_{t=1}^{T-1} x_t^T Q x_t + u_t^T R u_t$$

Subject to: $$x_{t+1} = f(x_t, u_t)$$

$$x_1 = \text{start state}$$

$$x_T = \text{goal state}$$

But, this is a nonlinear constraint – so how do we solve it now?

LTV (linear time varying) problem

Given: a nonlinear system: $x_{t+1} = f(x_t, u_t)$

A quadratic cost fn

A nominal trajectory: $x_1^*, u_t^*, \dots, x_{T-1}^*, u_{T-1}^*, x_T^*$

Find: a controller $u_t = -K_t x_t$ that works nearby nominal trajectory

Idea: time varying linear system

Linear Time Invariant (LTI):

$$x_{t+1} = Ax_t + Bu_t$$

$$J(X,U) = x_T^T Q_T x_T + \sum_{t=1}^{T-1} x_t^T Q x_t + u_t^T R u_t$$

Linear Time Variant (LTV):

Notice time time dependence
– each time step is linearized differently

$$x_{t+1} = \boxed{A_t} x_t + \boxed{B_t} u_t$$

$$J(X,U) = x_T^T Q_T x_T + \sum_{t=1}^{T-1} x_t^T \boxed{Q_t} x_t + u_t^T \boxed{R_t} u_t$$

Linear Time Varying LQR (LTV)

Similar first order taylor series expansion as before:

$$x_{t+1} \approx \underbrace{f(x_t^*, u_t^*)}_{} + \underbrace{\frac{\partial f}{\partial x}(x_t^*, u_t^*)}_{}(x_t - x_t^*) + \underbrace{\frac{\partial f}{\partial u}(x_t^*, u_t^*)}_{}(u_t - u_t^*)$$

$$= x_{t+1}^* + A_t(x_t - x_t^*) + B_t(u_t - u_t^*)$$

Linear Time Varying LQR (LTV)

Similar first order taylor series expansion as before:

$$x_{t+1} \approx \underbrace{f(x_t^*, u_t^*)}_{} + \underbrace{\frac{\partial f}{\partial x}(x_t^*, u_t^*)}_{}(x_t - x_t^*) + \underbrace{\frac{\partial f}{\partial u}(x_t^*, u_t^*)}_{}(u_t - u_t^*)$$

$$= x_{t+1}^* + A_t(x_t - x_t^*) + B_t(u_t - u_t^*)$$

Solution: solve LQR for this TV system: $\bar{x}_{t+1} = A_t \bar{x}_t + B_t \bar{u}_t$

$$\bar{x}_t = x_t - x_t^* \quad \bar{u}_t = u_t - u_t^*$$

Resulting controller is linearized about nominal trajectory: $\bar{u}_t^* = -K_t \bar{x}_t$

Linear Time Varying LQR (LTV)

Similar first order t

$$x_{t+1} \approx f(x_t^*$$

$$u_t^*)(u_t - u_t^*)$$

$$= x_{t+1}^*$$

Key points:

– point linearized about changes on each time step

– nominal trajectory does not even need to be feasible!

– you DO NEED to know what time step you're at throughout

Solution: solve LQR

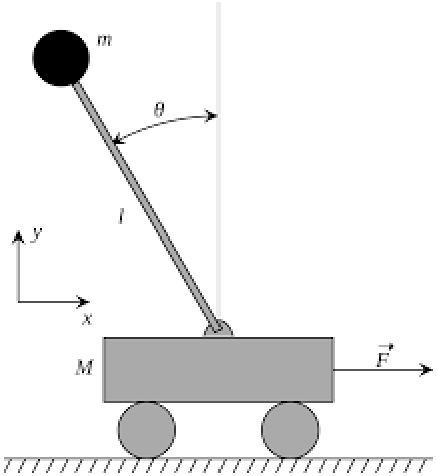
$$B_t \bar{u}_t$$

$$\bar{x}_t = x_t \qquad \bar{u}_t = u_t - u_t^*$$

Resulting controller is linearized about nominal trajectory: $\bar{u}_t^* = -K_t \bar{x}_t$

Question

Think about cart-pole



We have shown that you can use linearize LQR about an arbitrary trajectory.

Can you just linearize about the current operating point on each time step?

Back to nonlinear control problem...

Minimize: $$J(X,U) = x_T^T Q_T x_T + \sum_{t=1}^{T-1} x_t^T Q x_t + u_t^T R u_t$$

Subject to: $$x_{t+1} = f(x_t, u_t)$$
$$x_1 = \text{start state}$$
$$x_T = \text{goal state}$$

LTV LQR only works if you have a nominal trajectory

How do you get the nominal trajectory???

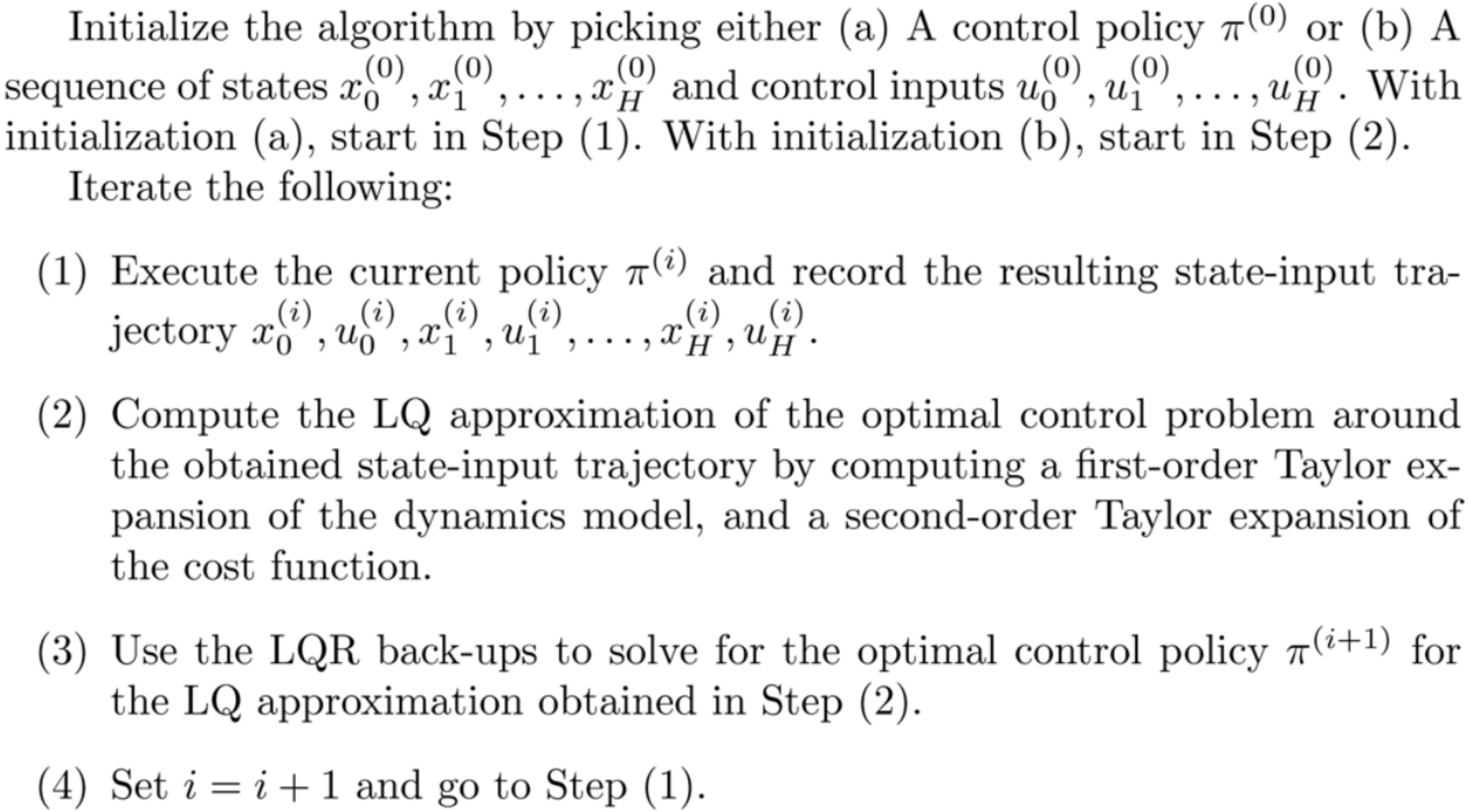
Iterative LQR

Key observation:

If: you start with a bad nominal trajectory, run LTV LQR linearized about it, and then integrate forward the resulting locally optimal policy…

Then: the resulting trajectory will be better (lower cost) than the nominal trajectory you started with

Iterative LQR



Initialize the algorithm by picking either (a) A control policy $\pi^{(0)}$ or (b) A sequence of states $x_0^{(0)}, x_1^{(0)}, \ldots, x_H^{(0)}$ and control inputs $u_0^{(0)}, u_1^{(0)}, \ldots, u_H^{(0)}$. With initialization (a), start in Step (1). With initialization (b), start in Step (2).

Iterate the following:

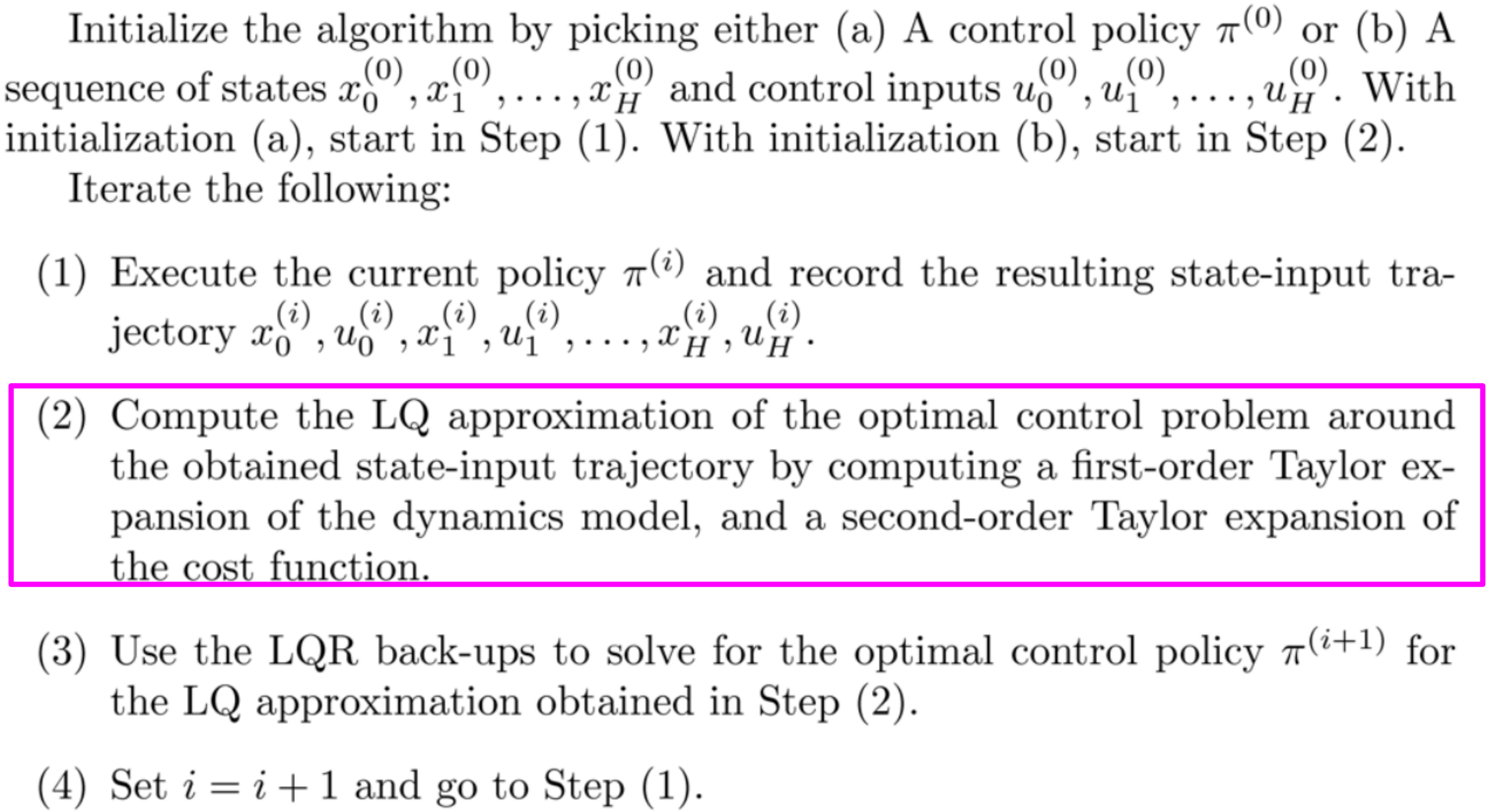
(1) Execute the current policy $\pi^{(i)}$ and record the resulting state-input trajectory $x_0^{(i)}, u_0^{(i)}, x_1^{(i)}, u_1^{(i)}, \ldots, x_H^{(i)}, u_H^{(i)}$.

(2) Compute the LQ approximation of the optimal control problem around the obtained state-input trajectory by computing a first-order Taylor expansion of the dynamics model, and a second-order Taylor expansion of the cost function.

(3) Use the LQR back-ups to solve for the optimal control policy $\pi^{(i+1)}$ for the LQ approximation obtained in Step (2).

(4) Set $i = i + 1$ and go to Step (1).

Iterative LQR



Initialize the algorithm by picking either (a) A control policy $\pi^{(0)}$ or (b) A sequence of states $x_0^{(0)}, x_1^{(0)}, \ldots, x_H^{(0)}$ and control inputs $u_0^{(0)}, u_1^{(0)}, \ldots, u_H^{(0)}$. With initialization (a), start in Step (1). With initialization (b), start in Step (2).

Iterate the following:

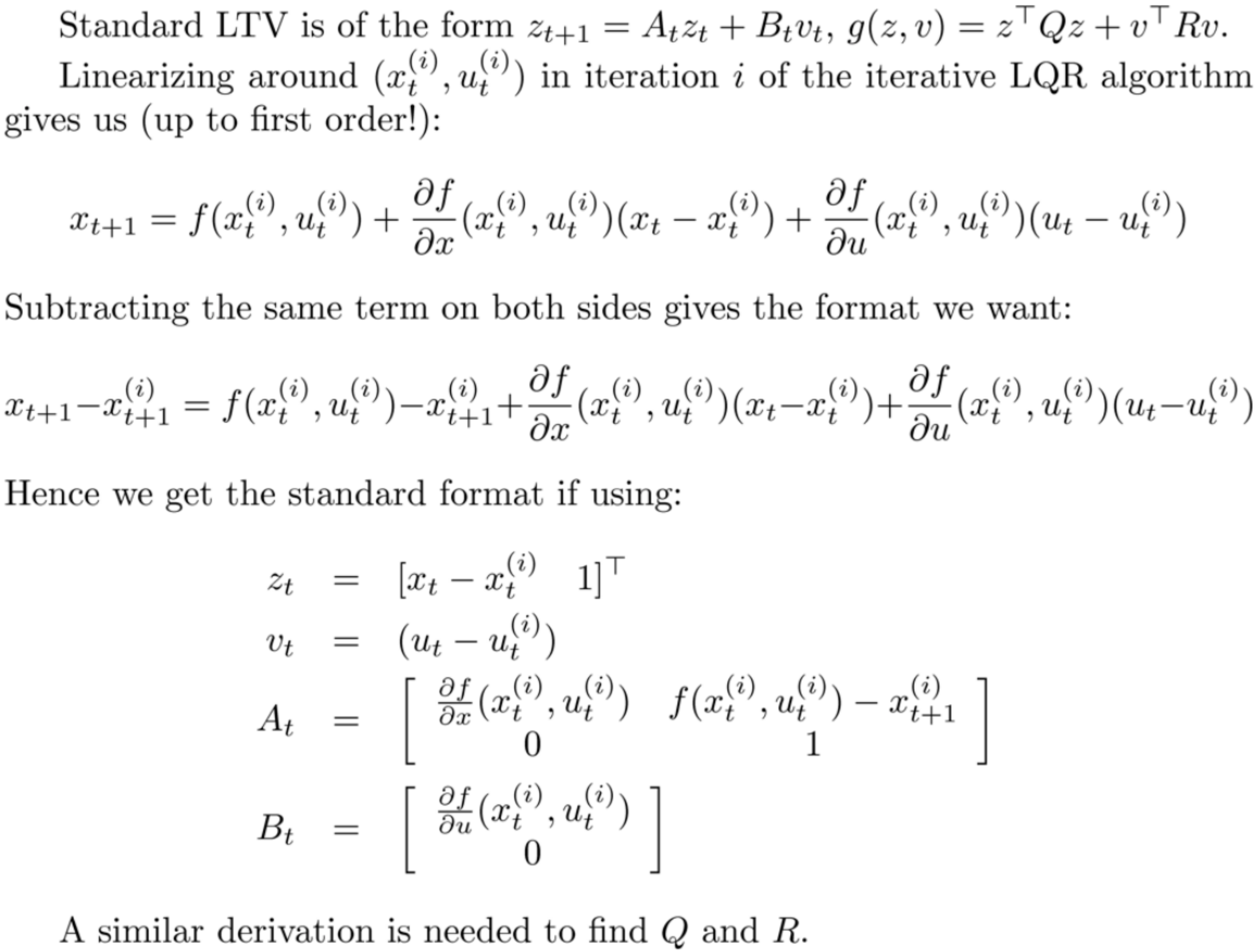
(1) Execute the current policy $\pi^{(i)}$ and record the resulting state-input trajectory $x_0^{(i)}, u_0^{(i)}, x_1^{(i)}, u_1^{(i)}, \ldots, x_H^{(i)}, u_H^{(i)}$.

(2) Compute the LQ approximation of the optimal control problem around the obtained state-input trajectory by computing a first-order Taylor expansion of the dynamics model, and a second-order Taylor expansion of the cost function.

(3) Use the LQR back-ups to solve for the optimal control policy $\pi^{(i+1)}$ for the LQ approximation obtained in Step (2).

(4) Set $i = i + 1$ and go to Step (1).

Iterative LQR



Standard LTV is of the form $z_{t+1} = A_t z_t + B_t v_t$, $g(z,v) = z^\top Q z + v^\top R v$.

Linearizing around $(x_t^{(i)}, u_t^{(i)})$ in iteration i of the iterative LQR algorithm gives us (up to first order!):

$$x_{t+1} = f(x_t^{(i)}, u_t^{(i)}) + \frac{\partial f}{\partial x}(x_t^{(i)}, u_t^{(i)})(x_t - x_t^{(i)}) + \frac{\partial f}{\partial u}(x_t^{(i)}, u_t^{(i)})(u_t - u_t^{(i)})$$

Subtracting the same term on both sides gives the format we want:

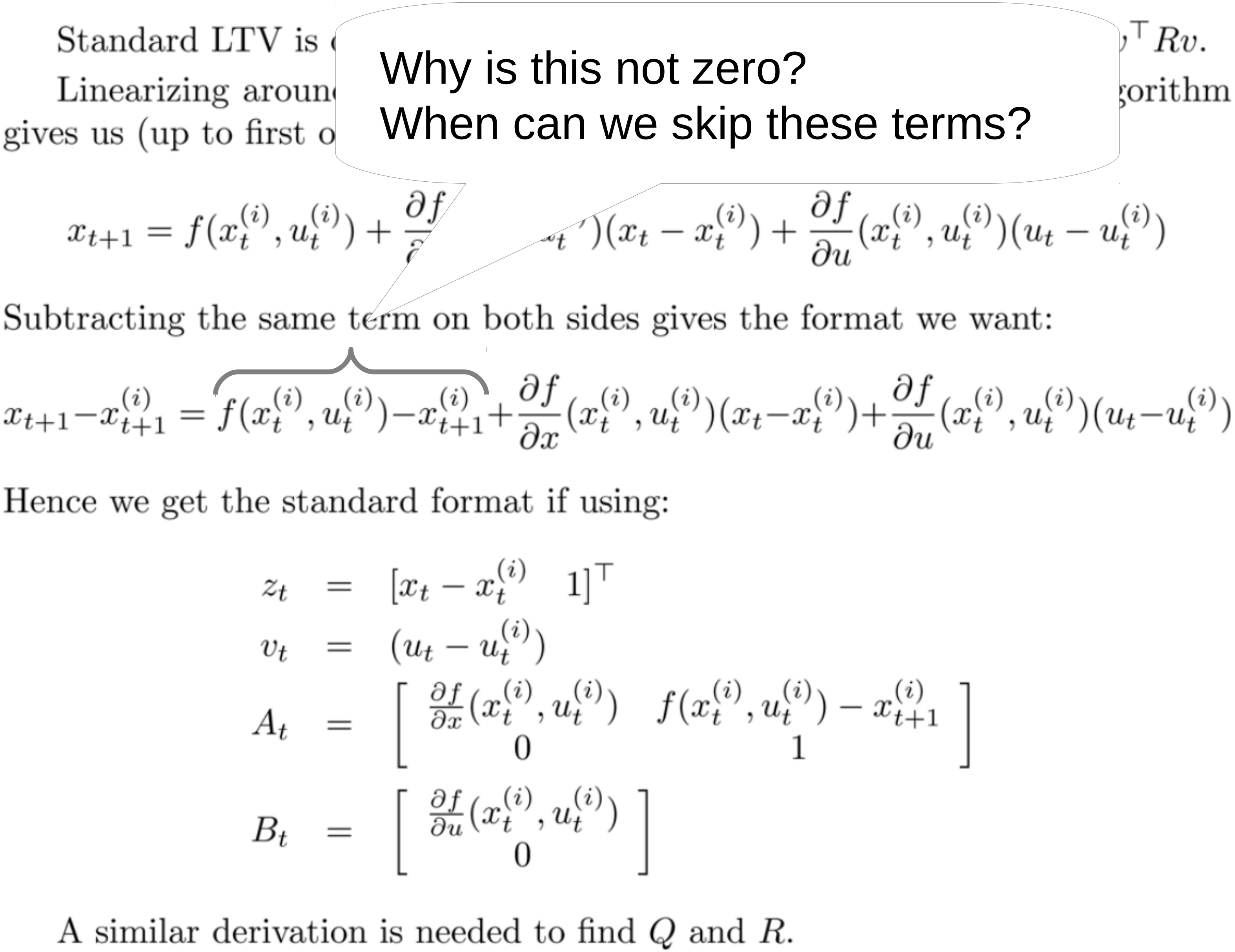
$$x_{t+1} - x_{t+1}^{(i)} = f(x_t^{(i)}, u_t^{(i)}) - x_{t+1}^{(i)} + \frac{\partial f}{\partial x}(x_t^{(i)}, u_t^{(i)})(x_t - x_t^{(i)}) + \frac{\partial f}{\partial u}(x_t^{(i)}, u_t^{(i)})(u_t - u_t^{(i)})$$

Hence we get the standard format if using:

$$\begin{aligned}
z_t &= [x_t - x_t^{(i)} \quad 1]^\top \\
v_t &= (u_t - u_t^{(i)}) \\
A_t &= \begin{bmatrix} \frac{\partial f}{\partial x}(x_t^{(i)}, u_t^{(i)}) & f(x_t^{(i)}, u_t^{(i)}) - x_{t+1}^{(i)} \\ 0 & 1 \end{bmatrix} \\
B_t &= \begin{bmatrix} \frac{\partial f}{\partial u}(x_t^{(i)}, u_t^{(i)}) \\ 0 \end{bmatrix}
\end{aligned}$$

A similar derivation is needed to find Q and R.

Iterative LQR



Standard LTV is ... $v^\top Rv$.

Linearizing aroun... gorithm

gives us (up to first o...

Why is this not zero?
When can we skip these terms?

$$x_{t+1} = f(x_t^{(i)}, u_t^{(i)}) + \frac{\partial f}{\partial} \ldots)(x_t - x_t^{(i)}) + \frac{\partial f}{\partial u}(x_t^{(i)}, u_t^{(i)})(u_t - u_t^{(i)})$$

Subtracting the same term on both sides gives the format we want:

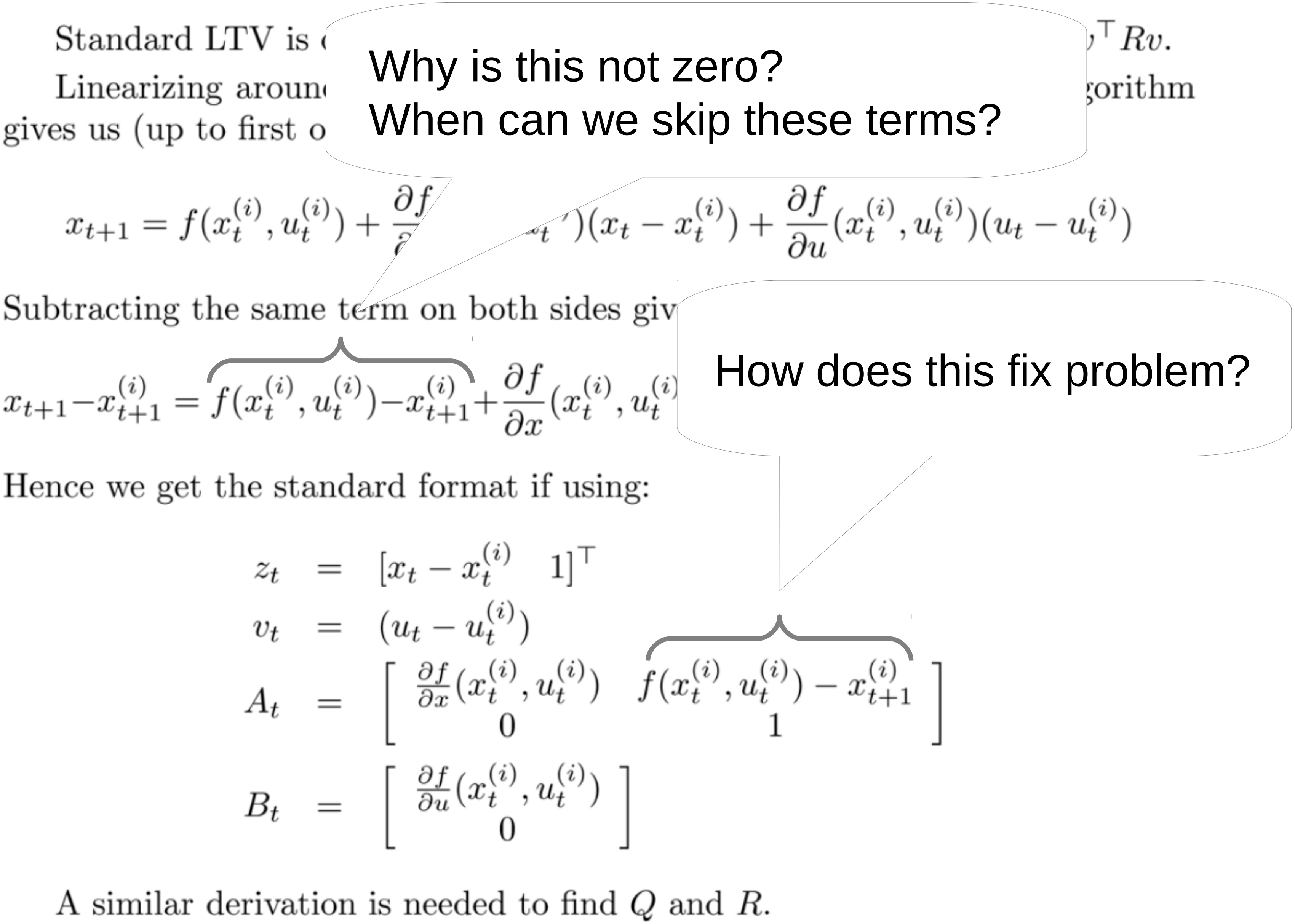
$$x_{t+1} - x_{t+1}^{(i)} = \overbrace{f(x_t^{(i)}, u_t^{(i)}) - x_{t+1}^{(i)}} + \frac{\partial f}{\partial x}(x_t^{(i)}, u_t^{(i)})(x_t - x_t^{(i)}) + \frac{\partial f}{\partial u}(x_t^{(i)}, u_t^{(i)})(u_t - u_t^{(i)})$$

Hence we get the standard format if using:

$$\begin{aligned}
z_t &= [x_t - x_t^{(i)} \quad 1]^\top \\
v_t &= (u_t - u_t^{(i)}) \\
A_t &= \begin{bmatrix} \frac{\partial f}{\partial x}(x_t^{(i)}, u_t^{(i)}) & f(x_t^{(i)}, u_t^{(i)}) - x_{t+1}^{(i)} \\ 0 & 1 \end{bmatrix} \\
B_t &= \begin{bmatrix} \frac{\partial f}{\partial u}(x_t^{(i)}, u_t^{(i)}) \\ 0 \end{bmatrix}
\end{aligned}$$

A similar derivation is needed to find Q and R.

Iterative LQR



Standard LTV is $v^\top Rv$.

Linearizing aroun gorithm gives us (up to first o

$$x_{t+1} = f(x_t^{(i)}, u_t^{(i)}) + \frac{\partial f}{\partial} \quad)(x_t - x_t^{(i)}) + \frac{\partial f}{\partial u}(x_t^{(i)}, u_t^{(i)})(u_t - u_t^{(i)})$$

Subtracting the same term on both sides giv

$$x_{t+1} - x_{t+1}^{(i)} = \overbrace{f(x_t^{(i)}, u_t^{(i)}) - x_{t+1}^{(i)}} + \frac{\partial f}{\partial x}(x_t^{(i)}, u_t^{(i)}$$

Hence we get the standard format if using:

$$\begin{aligned}
z_t &= [x_t - x_t^{(i)} \quad 1]^\top \\
v_t &= (u_t - u_t^{(i)}) \\
A_t &= \begin{bmatrix} \frac{\partial f}{\partial x}(x_t^{(i)}, u_t^{(i)}) & \overbrace{f(x_t^{(i)}, u_t^{(i)}) - x_{t+1}^{(i)}} \\ 0 & 1 \end{bmatrix} \\
B_t &= \begin{bmatrix} \frac{\partial f}{\partial u}(x_t^{(i)}, u_t^{(i)}) \\ 0 \end{bmatrix}
\end{aligned}$$

A similar derivation is needed to find Q and R.

Why is this not zero?
When can we skip these terms?

How does this fix problem?

Iterative LQR

- Need not converge as formulated!
  - Reason: the optimal policy for the LQ approximation might end up not staying close to the sequence of points around which the LQ approximation was computed by Taylor expansion.
  - Solution: in each iteration, adjust the cost function so this is the case, i.e., use the cost function

$$(1-\alpha)g(x_t, u_t) + \alpha(\|x_t - x_t^{(i)}\|_2^2 + \|u_t - u_t^{(i)}\|_2^2)$$

Assuming g is bounded, for α close enough to one, the 2nd term will dominate and ensure the linearizations are good approximations around the solution trajectory found by LQR.

Iterative LQR

- f is non-linear, hence this is a non-convex optimization problem. Can get stuck in local optima! Good initialization matters.

- g could be non-convex: Then the LQ approximation fails to have positive-definite cost matrices.
    - Practical fix: if Q_t or R_t are not positive definite → increase penalty for deviating from current state and input ($x^{(i)}_t$, $u^{(i)}_t$) until resulting Q_t and R_t are positive definite.

Iterative LQR

iLQR works well in the MPC context

– stabilization to the trajectory will handle most small deviations from nominal

– can iterate the process to handle larger deviations