Iterative Linearized Control: Stable Algorithms and Complexity Guarantees

Vincent Roulet, Siddhartha Srinivasa,
Dmitriy Drusvyatskiy, Zaid Harchaoui

ICML 2019

Problem

Nonlinear control $\longrightarrow$ **Iterative linearization (ILQR)**

around current x_t, u_t

$$\min_{\substack{u_0,\ldots,u_{T-1}\\ x_0,\ldots x_T}} \sum_{t=0}^{T} \Big(h_t(x_t) + g_t(u_t) \Big)$$

$$\text{s.t.} \quad \begin{aligned} x_{t+1} &= \phi_t(x_t, u_t) \\ x_0 &= \hat{x}_0 \end{aligned}$$

$$\min_{\substack{v_0,\ldots,v_{T-1}\\ y_0,\ldots y_T}} \sum_{t=0}^{T} \Big(y_t^\top H_t y_t + v_t^\top G_t v_t \Big)$$

$$\text{s.t.} \quad \begin{aligned} y_{t+1} &= \Phi_{t,x} y_t + \Phi_{t,u} v_t \\ y_0 &= 0 \end{aligned}$$

$\rightarrow$ Next iterate $u_t^+ = u_t + v_t^*$

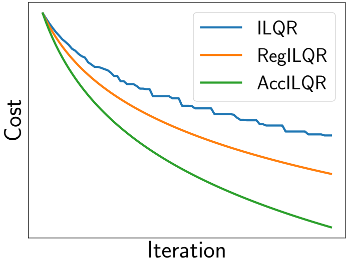
Questions

1. Does ILQR converge? Can it be accelerated?
2. How do we characterize complexities for nonlinear control?

Contributions

Regularized and Accelerated ILQR

1. ILQR is Gauss-Newton
   $\rightarrow$ **Regularized** ILQR gets convergence to a stationary point
2. Potential acceleration by extrapolation steps
   $\rightarrow$ **Accelerated** ILQR akin to Catalyst acceleration



Contributions

Oracles complexities

1. Oracles are solved by dynamic programming
   $\rightarrow$ Gradient and Gauss-Newton have **both** cost in $\mathcal{O}(T)$
2. Automatic-differentiation software libraries available
   $\rightarrow$ Use auto.-diff. as oracle for **direct** implementation

Code summary available at https://github.com/vroulet/ilqc

```
dynamics, cost = define_ctrl_pb()

ctrl = rand(dim_ctrl)

auto_diff_oracle = define_auto_diff_oracle(ctrl, dynamics)
dual_sol = sovle_dual_step(ctrl, cost, auto_diff_oracle)

next_ctrl = get_primal(dual_sol, auto_diff_oracle, cost)
```

Come see **Poster #39 in Pacific Ballroom!**