Complexity Bounds of Iterative Linear Quadratic Optimization Algorithms for Discrete Time Nonlinear Control

Vincent Roulet
Co-authors: Siddhartha Srinivasa, Maryam Fazel, Zaid Harchaoui

paper: https://arxiv.org/abs/2204.02322
code: https://github.com/vroulet/ilqc

3rd Biennial Meeting of SIAM Pacific Northwest Section
05/21/2022
Nonlinear Control Problems

**Nonlinear Control problem**
- Continuous time system \( \dot{x}(t) = f(x(t), u(t)) \)
- Minimize cost \( h(x(t), t) \) over \( t \in [0, T] \) for \( x(0) \) fixed by controlling the system through \( u(t) \)
- Discretize dynamics as \( x_{t+1} = f(x_t, u_t) \)
- Minimize costs \( h_t(x_t) \) over \( t \in \{0, \ldots, \tau\} \) for \( x_0 \) fixed with respect to \( u_0, \ldots, u_{\tau-1} \)

**Algorithms Principle**
Current controls \( u_0, \ldots, u_{\tau-1} \) with trajectory \( x_0, \ldots, x_\tau \)
1. Linearize dynamics \( f \) around \( x_t, u_t \)
2. Take quadratic approx. of the costs \( h_t \) around \( x_t \)
3. Solve resulting lin. quad. problem
4. Repeat from 1.
Autonomous Car Racing

Simple model of a car

\[ x = (z_x, z_y, \theta, v), \quad u = (\delta, a) \]
\[ \dot{z}_x = v \cos \theta \quad \dot{\theta} = v \tan(\delta) \]
\[ \dot{z}_y = v \sin \theta \quad \dot{v} = a \]

Algo. converges fast to optimal trajectory

Optimized trajectory horizon \( \tau = 100 \)

Convergence of the algorithm
Autonomous Car Racing

Bicycle model of a car  
(Liniger et al. 2015)

Models tire forces (highly non-linear)

Unclear whether the algorithm succeeded...

Optimized trajectory horizon $\tau = 100$
Outline

A Sufficient Condition for Global Convergence

Implementation and Convergence Analysis
Outline

A Sufficient Condition for Global Convergence

Implementation and Convergence Analysis
Continuous Time Control problem

\[
\min_{x(t), u(t)} \int_0^T h(x(t), t) \, dt
\]

s.t. \quad \dot{x}(t) = f(x(t), u(t)), \quad x(0) = \bar{x}_0

Discrete Time Control Problem

\[
\min_{x_0, \ldots, x_\tau} \sum_{t=1}^\tau h_t(x_t)
\]

s.t. \quad x_{t+1} = f(x_t, u_t), \quad x_0 = \bar{x}_0

**Objective** for \( x_0 = \bar{x}_0, \ u = (u_0; \ldots; u_{\tau-1}) \ x = (x_1, \ldots, x_\tau) \)

\[
\mathcal{J}(u) = h(f^{[\tau]}(x_0, u)) \quad \text{for} \quad f^{[\tau]}(x_0, u) = (x_1; \ldots; x_\tau), \quad h(x) = \sum_{t=1}^\tau h_t(x_t)
\]

s.t. \quad x_{t+1} = f(x_t, u_t)

\[\text{Dynamics}\]

\[\text{Costs}\]

\[\mathcal{J}(u)\]
A Sufficient Condition for Global Convergence

Idea:
- Prove sufficient condition for global conv.
  of 1st order methods, such as, for $c > 0$,

$$\|\nabla \mathcal{J}(u)\|_2^2 \geq c(\mathcal{J}(u) - \mathcal{J}^*)$$

Gradient dominated objective $\mathcal{J}$

Derivation:
- Here consider that the total cost $h$ is e.g. $\mu$-strongly convex s.t.

$$\|\nabla h(x)\|_2^2 \geq \mu(h(x) - h^*)$$

- We have $\mathcal{J}(u) = h(f^{[\tau]}(x_0, u))$ so $\|\nabla \mathcal{J}(u)\|_2^2 = \|\nabla_u f^{[\tau]}(x_0, u) \nabla h(x)\|_2^2$

- So if $f^{[\tau]}(x_0, u)$ satisfies

$$\forall u \quad \sigma(\nabla_u f^{[\tau]}(x_0, u)) := \inf_{\lambda} \frac{\|\nabla_u f^{[\tau]}(x_0, u) \lambda\|_2}{\|\lambda\|_2} \geq \sigma > 0$$

where $\sigma_{\text{min}}(A) = \inf_{\|z\| > 0} \frac{\|Az\|_2}{\|z\|_2}$ is the minimal singular value of $A$

then

$$\|\nabla \mathcal{J}(u)\|_2^2 \geq \sigma^2 \|\nabla h(x)\|_2^2 \geq \sigma^2 \mu(h(x) - h^*) = \sigma^2 \mu(\mathcal{J}(u) - \mathcal{J}^*)$$
Interpretation of a Sufficient Condition for Global Convergence

Interpretation

$$\sigma(\nabla_u f^{[\tau]}(x_0, u)) > 0$$

$$\iff$$ Reverse mode of auto-diff $$\lambda \mapsto \nabla_u f^{[\tau]}(x_0, u)\lambda$$ is injective

$$\iff$$ Forward mode of auto-diff $$v \mapsto \nabla_u f^{[\tau]}(x_0, u)^\top v$$ is surjective

Here $$y = \nabla_u f^{[\tau]}(x_0, u)^\top v$$ is the linearization of the trajectories given as

$$y_{t+1} = \nabla_{x_t} f(x_t, u_t)^\top y_t + \nabla_{u_t} f(x_t, u_t)^\top v_t, \quad y_0 = 0$$

So $$\sigma(\nabla_u f^{[\tau]}(x_0, u)) > 0$$ if the linearization of the trajectories are surjective

How to verify this condition from $$f$$ only?

Previous work
Sufficient optimality conditions in continuous time done by Mangasarian (1966)

$$\rightarrow$$ Translatable in discrete time but requires convexity of implicitly defined functions...
Lemma (R. et al. (2022))

If the linearization, \( v \rightarrow \nabla_u f(x, u) \top v \), of \( l_f \)-Lip. cont. dynamics \( f \) is surjective,

\[
\forall x, u, \quad \sigma(\nabla_u f(x, u)) \geq \sigma_f > 0,
\]

then the linearization of the trajectories, \( v \rightarrow \nabla_u f[\tau](x_0, u) \top v \), is surjective,

\[
\forall x_0, u, \quad \sigma(\nabla_u f[\tau](x_0, u)) \geq \frac{\sigma_f}{1 + l_f} > 0,
\]

Problem:

- Usually less control variables than state variables \( \dim(u(t)) < \dim(x(t)) \)
  
  So \( \sigma_{\min}(\nabla_u f(x(t), u(t))) > 0 \) impossible

→ Use multistep schemes
Intuition for a Sufficient Condition for Global Convergence

Pendulum dynamics

\[ m\ddot{\theta}(t) = -mg \sin \theta(t) - \mu \dot{\theta}(t) + u(t) \]

One step Euler scheme

\[ f(x_t, u_t) = x_{t+1} \text{ for } x_t = (\theta_t, \omega_t) \text{ with } \omega = \dot{\theta} \]

angle \quad \theta_{t+1} = \theta_t + \Delta \omega_t

angle speed \quad \omega_{t+1} = \omega_t - \Delta(g \sin \theta_t - \mu \omega_t) + \Delta u_t

Linearization surjective? \( \times \)

Two steps Euler scheme \( f(x_t, u_t) = x_{t+1} \) with \( u_t = (v_t, v_{t+1/2}) \)

\[ \begin{align*}
\theta_{t+1/2} &= \theta_t + \Delta \omega_t \\
\omega_{t+1/2} &= \omega_t - \Delta(g \sin \theta_t - \mu \omega_t) + \Delta v_t
\end{align*} \]

\[ \begin{align*}
\theta_{t+1} &= \theta_t + \ldots + \Delta^2 v_t \\
\omega_{t+1} &= \omega_t + \ldots + \Delta v_{t+1/2}
\end{align*} \]

Linearization surjective w.r.t. \( u_t = (v_t, v_{t+1/2}) \) ? \( \checkmark \)
Overall Analysis

**Trajectory** decomposed in $\tau$ steps

$$f^{[\tau]}(x_0, u) = (x_1; \ldots; x_\tau)$$

s.t. $x_{t+1} = f(x_t, u_t)$

**Dynamic** fractionated in $k$ steps

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

s.t. $x_{t+(s+1)/k} = \phi(x_{t+s/k}, u_{t+s/k})$

such as $\phi(y_t, v_t) = y_t + \Delta f(y_t, v_t)$ for $f$ continuous-time dynamic.

Sufficient condition for global convergence can be verified by analyzing whether $\phi$ can be *linearized by static feedback*, see R. et al. (2022)
Outline

A Sufficient Condition for Global Convergence

Implementation and Convergence Analysis
Implementation

Gradient oracle
- Linear approx. of dynamics, costs,
- Gradients of objective computed through dynamics

Forward pass
Compute objective and linear approx.
**Implementation**

**Gradient oracle**
- Linear approx. of dynamics, costs,
- Gradients of objective computed through dynamics

**Backward pass**
Backpropagate gradients through Matrix Vector Products (MVP)
Output gradients of objective w.r.t. control variables
Implementation

Linear Quadratic Regulator oracle
- Linear approx. of dynamics, quadratic approx. of costs
- Regularized linear quadratic approx. of objective minimized through dynamics

Forward pass
Compute objective, linear approx. of dynamics, quad. approx. of costs
Implementation

**Linear Quadratic Regulator oracle**

- Linear approx. of dynamics, quadratic approx. of costs
- Regularized linear quadratic approx. of objective minimized through dynamics

**Backward pass**

Define recursively minimum cost of reg. lin. quad. approx. starting from any \( y_t \) at time \( t \)

\[
c_t : y_t \mapsto q_{h_t}^{x_t}(y_t) + \min_{v_t} \left\{ \nu \|v_t\|_2^2 + c_{t+1}(\ell^{x_t,u_t}_f(y_t,v_t)) \right\}
\]

- \( q_{h_t}^{x_t} \) quad. approx. of \( h_t \) on \( x_t \)
- \( \ell^{x_t,u_t}_f \) lin. approx. of \( f \) on \( x_t, u_t \)

where \( c_t \) is a quad. function param. by Matrices Inverse & Matrices Products (MIMP)
Implementation

**Linear Quadratic Regulator oracle**
- Linear approx. of dynamics, quadratic approx. of costs
- Regularized linear quadratic approx. of objective minimized through dynamics

**Backward pass**
Record optimal control of reg. lin. quad. approx. starting from any $y_t$ at time $t$

$$\pi_t : y_t \mapsto \arg\min_{v_t} \left\{ \nu \|v_t\|_2^2 + c_{t+1} \left( \ell_{f,x_t,u_t} (y_t, v_t) \right) \right\}$$
Implementation

Linear Quadratic Regulator oracle
- Linear approx. of dynamics, quadratic approx. of costs
- Regularized linear quadratic approx. of objective minimized through dynamics

Roll-out pass
Roll-out optimal controls along the linear dynamics

$$v_t = \pi_t(y_t), \quad y_{t+1} = \ell^{x_t,u_t}(y_t,v_t)$$
Convergence Analysis

Problem

\[
\min_u \{ J(u) = h(g(u)) \}, \text{ where } g(u) = f^{[\tau]}(\bar{x}_0, u), \quad h(x) = \sum_{t=1}^{\tau} h_t(x_t)
\]

Algorithm (Li & Todorov 2007)

\[
u^{(k+1)} = \nu^{(k)} + LQR_{\nu_k}(J)(\nu^{(k)}) \quad \text{(ILQR)}
\]

where \( LQR_{\nu_k}(J)(\nu^{(k)}) \) is the oracle returning a direction computed by dynamic programming with a regularization \( \nu_k \)

Assumptions

- costs \( h_t \): \( \mu_h \)-strongly convex, \( L_h \)-smooth, \( M_h \)-smooth Hessian
  \( \rightarrow \) same for overall cost \( h \)

- dynamic \( f \): \( l_f \)-Lip. continuous, \( L_f \) smooth with \( \sigma(\nabla_u f(x, u)) \geq \sigma_f > 0 \)
  \( \rightarrow \) mapping \( g \): \( l_g \)-Lip.continuous, \( L_g \)-smooth with \( \sigma(\nabla g(u)) \geq \sigma_g > 0 \)
  with \( l_g, L_g, \sigma_g \) estimable from \( l_f, L_f, \sigma_f \)
Convergence Analysis Viewpoint

**ILQR as a generalized Gauss-Newton** (Sideris & Bobrow 2005)

- Overall ILQR minimizes a quadratic approx. of $h$ on top of a linear approx. of $g$
- So it can be summarized as

$$LQR_\nu(\mathcal{J})(u) = \arg\min_v q_h^g(u)(\ell_g^u(v)) + \frac{\nu}{2} \|v\|_2^2$$

$$= -\nabla g(u)\nabla^2 h(g(u))\nabla g(u)^\top + \nu I)^{-1}\nabla g(u)\nabla h(g(u))$$

which is a **regularized generalized Gauss-Newton method**

**Convergence proof idea**

1. For large enough regularization, $LQR_\nu(\mathcal{J})(u) \approx -\nu^{-1}\nabla g(u)\nabla h(g(u))$
   $\rightarrow$ linear global convergence possible as for a gradient descent

2. Denoting $x^{\text{next}} = g(u + v)$ for $v = LQR_\nu(\mathcal{J})(u)$, with simple linear algebra,

$$x^{\text{next}} \approx g(u) + \nabla g(u)^\top v = x - (\nabla^2 h(x) + \nu(\nabla g(u)^\top \nabla g(u))^{-1})^{-1}\nabla h(x).$$

so for small enough regularization $x^{\text{next}} \approx x - \nabla^2 h(x)^{-1}\nabla h(x)$

$\rightarrow$ local quadratic convergence possible as for a Newton method

3. Can show that a regularization $\nu \propto \|\nabla h(x)\|_2$ ensures both!

**Previous work**

- Global convergence of **regularized** Gauss-Newton a.k.a. Levenberg-Marquardt e.g. (Bergou et al. 2020)
Theorem (R. et al. (2022))

Under the aforementioned assumptions, the ILQR algorithm equipped with $\nu(u) = \bar{\nu} \|\nabla h(g(u))\|_2$ for $\bar{\nu}$ large enough converges to accuracy $\varepsilon$ in

\[
\begin{align*}
4\theta_g (\sqrt{\delta_0} - \sqrt{\delta}) & + 2\rho_h \ln \left( \frac{\delta_0}{\delta} \right) + 2\alpha \ln \left( \frac{\theta_g \sqrt{\delta_0} + \rho_g}{\theta_g \sqrt{\delta} + \rho_g} \right) + O(\ln \ln(\varepsilon)) \\
\text{1st phase} & \hspace{10em} \text{2nd phase} & \hspace{10em} \text{3rd phase}
\end{align*}
\]

iterations, each having a comput. complexity $O(\tau(\dim(x) + \dim(u))^3)$, where

- $\delta_0 = J(u^{(0)}) - J^*$ is the initial gap
- $\delta$ is the gap of quadratic conv. $: \delta_0 \leq \delta \implies 3rd$ phase
- $\rho_h = L_h/\mu_h$ is the condition number of the costs
- $\rho_g = I_g/\sigma_g$ is the condition number of the linearized traj.
- $\theta_h = M_h/\mu_h^{3/2}$ is the param. of self-concordance of the costs
- $\theta_g = L_g/(\sigma_g^2 \sqrt{\mu_h})$ acts as self-concordance param. for the linear-quadratic decomp.
- $\alpha$ is another cond. nb

---

1Extensions to self-concordant or gradient dominated costs, differential dynamic programming algorithms available
import torch
from envs.car import Car
from envs.backward import lin_quad_backward, quad_backward
from envs.rollout import roll_out_lin

# Define control problem and candidate control variables
env = Car(model='simple', discretization='euler', cost='exact',
          horizon=50, dt=0.02)
ctrls = torch.randn(env.horizon, env.dim_ctrl, requires_grad=True)

# ILQR/Gauss-Newton step
traj, costs = env.forward(ctrls, approx='linquad')
policies = lin_quad_backward(traj, costs, reg_ctrl=1.)[0]
gauss_newton_dir = roll_out_lin(traj, policies)
gauss_newton_step = ctrls + gauss_newton_dir

# Newton and Differentiable Dynamic Programming also available
Conclusion

Summary

- Conv. guarantees for canonical noncvx pb → analyze problem at elementary scale as done in a diff. prog. implementation
- Complexity bounds for ILQR → quad. convergence at low iteration cost by using a diff. prog. implementation

Model Predictive Control & contouring objective

Thank you for your attention!


