ECE586BH: Interplay between Control and Machine Learning

Bin Hu

ECE, University of Illinois Urbana-Champaign

Lecture 1, Fall 2023
Feedback Control

- dynamical systems
- robustness
- safety-critical
- model-based design
- CDC/ACC/ECC

Machine learning

- statistics/optimization
- large-scale (big data)
- performance-driven
- train using data
- NeurIPS/ICML/ICLR
Artificial Intelligence Revolution

Google Translate
Break through language barriers

Safety-critical applications!
Flight Control Certification

Unified and automated tools for a repeatable and trustable design process of next generation intelligent systems
Example: Robustness is crucial!

- Deep learning: Small adversarial perturbations can fool the classifier!

\[ x_{k+1} = x_k - \alpha (\nabla f(x_k) + e_k) \]

- Optimization: The oracle can be inexact!

- Decision and control: Model uncertainty and sim-to-real gap matter!
Control for Learning

Control theory addresses unified analysis and design of dynamical systems.

<table>
<thead>
<tr>
<th>LTI systems</th>
<th>Markov jump systems</th>
<th>Lur’e systems</th>
</tr>
</thead>
<tbody>
<tr>
<td>$\xi_{k+1} = A\xi_k + Bu_k$</td>
<td>$\xi_{k+1} = A_{i_k}\xi_k + B_{i_k}u_k$</td>
<td>$\xi_{k+1} = A\xi_k + B\phi(C\xi_k)$</td>
</tr>
<tr>
<td>$A^TPA - P &lt; 0$</td>
<td>$\sum_{i=1}^{n} p_{ij} A_i^TP_iA_i &lt; P_j$</td>
<td>$\begin{bmatrix} A^TPA - P &amp; A^TPB \ B^TPA &amp; B^TPB \end{bmatrix} &lt; M$</td>
</tr>
</tbody>
</table>

**Pros:** Unified testing conditions when problem parameters are changed.

**Cons:** For control, we only need to solve the conditions numerically.

**Control for learning:** Algorithms and networks treated as control systems

- Neural networks as generalized Lur’e systems
- Stochastic learning algorithms as generalized Lur’e systems

**Key message:** Robustness can be addressed in a unified manner!
Control theory addresses unified analysis and design of dynamical systems.

<table>
<thead>
<tr>
<th>LTI systems</th>
<th>MJLS</th>
<th>Lur’e systems</th>
</tr>
</thead>
<tbody>
<tr>
<td>$\xi_{k+1} = A\xi_k + Bu_k$</td>
<td>$\xi_{k+1} = A_{i_k}\xi_k + B_{i_k}u_k$</td>
<td>$\xi_{k+1} = A\xi_k + B\phi(C\xi_k)$</td>
</tr>
<tr>
<td>$A^T PA - P &lt; 0$</td>
<td>$\sum_{i=1}^{n} p_{ij} A_i^T P_i A_i &lt; P_j$</td>
<td>$[A^T PA - P \ A^TPB] \prec M$</td>
</tr>
</tbody>
</table>

Many control design methods rely on convex conditions (BGFB1994). What about problems that cannot be formulated as convex optimization?

- Direct policy search (e.g. $\min J(K)$) is nonconvex!

Learning for control: Tailoring nonconvex learning theory to push robust control theory beyond the convex regime.
Outline

• Control for Learning
  • Control methods for certifiably robust neural networks
  • A control perspective on stochastic learning algorithms

• Learning for Control
  • Global convergence of direct policy search on robust control
Robust Control Theory

1. Approximate the true system as “a linear system + a perturbation”
2. $\Delta$ can be a troublesome element: nonlinearity, uncertainty, or delays
3. Rich control literature including standard textbooks
   - Zhou, Doyle, Glover, “Robust and optimal control,” 1996
4. Many tools: small gain, passivity, dissipativity, Zames-Falb multipliers, etc
5. The integral quadratic constraint (IQC) framework [Megretski, Rantzer (TAC1997)] provides a unified analysis for “LTI $P +$ troublesome $\Delta$”
6. Recently, IQC analysis has been extended for more general $P$
7. Typically, the stability is tested by a SDP condition
Quadratic Constraints from Robust Control

- **Lur’e system**: $\xi_{k+1} = A\xi_k + B\Delta(C\xi_k)$.
- **EX**: Gradient method ($A = I$, $B = -\alpha I$, $C = I$, and $\Delta = \nabla f$)
- **Question**: How to prove that the above Lur’e system converges? We are looking at the following set of coupled sequences $\{\xi_k, w_k, v_k\}$

$$\{ (\xi, w, v) : \xi_{k+1} = A\xi_k + Bw_k, v_k = C\xi_k \} \cap \{ (\xi, w, v) : w_k = \Delta(v_k) \}$$

- **Key idea**: Quadratic constraints! Replace the troublesome nonlinear element $\Delta$ with the following quadratic constraint:

$$\{ (v, w) : w_k = \Delta(v_k) \} \subset \left\{ (v, w) : \begin{bmatrix} v_k \\ w_k \end{bmatrix}^T M \begin{bmatrix} v_k \\ w_k \end{bmatrix} \leq 0 \right\},$$

where $M$ is constructed from the property of $\Delta$.

- If we can show that any sequence from the set below converges,

$$\left\{ (\xi, w, v) : \xi_{k+1} = A\xi_k + Bw_k, v_k = C\xi_k, \begin{bmatrix} v_k \\ w_k \end{bmatrix}^T M \begin{bmatrix} v_k \\ w_k \end{bmatrix} \leq 0 \right\},$$

then we are done.
Quadratic Constraints from Robust Control

Now we are analyzing the sequence from the following set:

\[
\{(\xi, w, v) : \xi_{k+1} = A\xi_k + Bw_k, \ v_k = C\xi_k, \begin{bmatrix} v_k \\ w_k \end{bmatrix}^T M \begin{bmatrix} v_k \\ w_k \end{bmatrix} \leq 0 \}
\]

**Theorem**

If there exists a positive definite matrix \(P\) and \(0 < \rho < 1\) s.t.

\[
\begin{bmatrix} A^TPA - \rho^2P & A^TPB \\ B^TPA & B^TPB \end{bmatrix} \preceq \begin{bmatrix} C & 0 \\ 0 & I \end{bmatrix}^T M \begin{bmatrix} C & 0 \\ 0 & I \end{bmatrix}
\]

then \(\xi_{k+1}^TP\xi_{k+1} \leq \rho^2\xi_k^TP\xi_k\) and \(\lim_{k \to \infty} \xi_k = 0\).

This condition is a semidefinite program (SDP) problem!
Illustrative Example: Gradient Descent Method

- Rewrite the gradient method $x_{k+1} = x_k - \alpha \nabla f(x_k)$ as:

$$
\xi_{k+1} = (x_{k+1} - x^*) = (x_k - x^*) - \alpha \nabla f(x_k) = \xi_k - \alpha \nabla f(x_k)
$$

- If $f$ is $L$-smooth and $m$-strongly convex, then by co-coercivity:

$$
\begin{bmatrix}
    x - x^* \\
    \nabla f(x)
\end{bmatrix}^T
\begin{bmatrix}
    2mLI & -(m+L)I \\
    -(m+L)I & 2I
\end{bmatrix}
\begin{bmatrix}
    x - x^* \\
    \nabla f(x)
\end{bmatrix} \leq 0
$$

- We have $A = I$, $B = -\alpha I$, $C = I$, and the following SDP

$$
\begin{bmatrix}
    A^T PA - \rho^2 P & A^T PB \\
    B^T PA & B^T PB
\end{bmatrix} \preceq
\begin{bmatrix}
    C & 0 \\
    0 & I
\end{bmatrix}^T M
\begin{bmatrix}
    C & 0 \\
    0 & I
\end{bmatrix}
$$

- This leads to

$$
\left(\begin{bmatrix}
    (1 - \rho^2)p & -\alpha p \\
    -\alpha p & \alpha^2 p
\end{bmatrix} + \begin{bmatrix}
    -2mL & m + L \\
    m + L & -2
\end{bmatrix}\right) \otimes I \preceq 0
$$

- Choose $(\alpha, \rho, p)$ to be $(\frac{1}{L}, 1 - \frac{m}{L}, L^2)$ or $(\frac{2}{L+m}, \frac{L-m}{L+m}, \frac{1}{2}(L+m)^2)$ to recover standard rates, i.e. $\|x_{k+1} - x^*\| \leq (1 - m/L)\|x_k - x^*\|$

- For this proof, is strong convexity really needed? No! Regularity condition!
Illustrative Example: Gradient Descent Method

- We have shown \( \|x_{k+1} - x^*\| \leq (1 - m/L)\|x_k - x^*\| \)
- Is it a contraction, i.e. \( \|x_{k+1} - x'_{k+1}\| \leq (1 - m/L)\|x_k - x'_k\| ? \)
- \( (x_{k+1} - x'_{k+1}) = (x_k - x'_k) - \alpha (\nabla f(x_k) - \nabla f(x'_k)) \)
- If \( f \) is \( L \)-smooth and \( m \)-strongly convex, then by co-coercivity:

\[
\begin{bmatrix}
x - x' \\
\nabla f(x) - \nabla f(x')
\end{bmatrix}^T
\begin{bmatrix}
2mLI & -(m + L)I \\
-(m + L)I & 2I
\end{bmatrix}
\begin{bmatrix}
x - x' \\
\nabla f(x) - \nabla f(x')
\end{bmatrix} \leq 0
\]

- We have \( A = I, B = -\alpha I, C = I \), and the same SDP

\[
\left( \begin{bmatrix}
(1 - \rho^2)p & -\alpha p \\
-\alpha p & \alpha^2 p
\end{bmatrix} + \begin{bmatrix}
-2mL & m + L \\
 m + L & -2
\end{bmatrix} \right) \otimes I \preceq 0
\]

- Choose \( (\alpha, \rho, p) \) to be \( \left( \frac{1}{L}, 1 - \frac{m}{L}, L^2 \right) \) or \( \left( \frac{2}{L+m}, \frac{L-m}{L+m}, \frac{1}{2}(L + m)^2 \right) \) to give the contraction result!

- For this proof, is strong convexity really needed? Yes!
Outline

- Control for Learning
  - Control methods for certifiably robust neural networks
  - A control perspective on stochastic learning algorithms

- Learning for Control
  - Global convergence of direct policy search on robust control
Deep Learning for Classification

Deep learning has revolutionized the fields of AI and computer vision!

- Input space $\mathcal{X} \subset \mathbb{R}^d$ to a label space $\mathcal{Y} := \{1, \ldots, H\}$.
- Predict labels from image pixels
- Neural network classifier function $f := (f_1, \ldots, f_H) : \mathcal{X} \rightarrow \mathbb{R}^H$ such that the predicted label for an input $x$ is $\arg \max_j f_j(x)$.
- Input-label $(x, y)$ is correctly classified if $\arg \max_j f_j(x) = y$. 

![Diagram of a neural network with input images of a cat and a dog](image)

**Output**

Cat
Deep learning models: $f(x) = x_{D+1}$ and $x_0 = x$

- **Feedforward:** $x_{k+1} = \sigma(W_k x_k + b_k)$ for $k = 0, 1, \cdots, D$
- **Residual network:** $x_{k+1} = x_k - \sigma(W_k x_k + b_k)$ for $k = 0, 1, \cdots, D$
- **Many other structures:** transformers, etc

Deep learning models are expressive and generalize well, achieving state-of-the-art results in computer vision and natural language processing. However, ...
Adversarial Attacks and Robustness

*For correct labels (i.e. $\arg \max_j f_j(x) = y$), one may find $\|\tau\| \leq \varepsilon$ s.t. $\arg \max_j f_j(x + \tau) \neq y$ (small perturbation lead to wrong prediction)*

- Small perturbation can fool modern deep learning models!
- How to deploy deep learning models into safety-critical applications?
- **Certified robustness**: A classifier $f$ is *certifiably robust at radius* $\varepsilon \geq 0$ at point $x$ with label $y$ if for all $\tau$ such that $\|\tau\| \leq \varepsilon$ : $\arg \max_j f_j(x + \tau) = y$
1-Lipschitz Networks for Certified Robustness

- Tsuzuku, Sato, Sugiyama (NeurIPS2018): Let $f$ be $L$-Lipschitz. If we have
  \[ M_f(x) := \max(0, f_y(x) - \max_{y' \neq y} f_{y'}(x)) > \sqrt{2}L \varepsilon \]
  then we have for every $\tau$ such that $\|\tau\|_2 \leq \varepsilon$: $\arg\max_j f_j(x + \tau) = y$
- Perturbation smaller than $M_f(x)/\sqrt{2}L$ cannot deceive $f$ for datapoint $x$!
- If each layer of a network is 1-Lipchitz, the entire network is 1-Lipschitz.
- For each data point, we test whether $M_f(x) > \sqrt{2} \varepsilon$, and then count the percentage of data points that is guaranteed to be guarded for perturbation smaller than $\varepsilon$ (which is the certified accuracy for that $\varepsilon$).
- We need to train a Lipschitz neural network with good prediction margins!

Previous approaches:

- Spectral normalization (MKKY2018): $x_{k+1} = \sigma \left( \frac{1}{\|W_k\|_2} W_k x_k + b_k \right)$
- Orthogonality (TK2021, SF2021): $x_{k+1} = \sigma(W_k x_k + b_k)$ with $W_k^T W_k = I$
- Convex potential layer (MDAA2022): $x_{k+1} = x_k - \frac{2}{\|W_k\|^2} W_k \sigma(W_k^T x + b_k)$
- AOL (PL2022): $x_{k+1} = \sigma(W_k \text{diag}(\sum_j |W_k^T W_k|_{ij})^{-\frac{1}{2}} x_k + b_k)$
My Focus: Principles for 1-Lipschitz Networks

**Theorem (AHDAH2023)**

*If there exists nonsingular diagonal \(T_k\) s.t. \(W_k^T W_k \preceq T_k\), then we have*

1. **The layer** \(x_{k+1} = \sigma(W_k T_k^{-\frac{1}{2}} x_k + b_k)\) **is 1-Lipschitz for any 1-Lipschitz** \(\sigma\).
2. **The layer** \(x_{k+1} = x_k - 2W_k T_k^{-1} \sigma(W_k^T x + b_k)\) **is 1-Lipschitz if** \(\sigma\) **is ReLU.**

\[
\|x_{k+1} - x'_{k+1}\|^2 \leq \|W_k T_k^{-\frac{1}{2}} (x_k - x'_k)\|^2 = (x_k - x'_k)^T T_k^{-\frac{1}{2}} W_k^T W_k T_k^{-\frac{1}{2}} (x_k - x'_k) \\ \leq \|x_k - x'_k\|^2
\]

The second statement can be proved using the quadratic constraint argument.

**A Unification of Existing 1-Lipschitz Neural Networks**

- **Spectral normalization:** Statement 1 with \(T_k = \|W_k\|^2_2 I\)
- **Orthogonal weights:** Statement 1 with \(T_k = I\) and \(W_k^T W_k = I\)
- **CPL:** Statement 2 with \(T_k = \|W_k\|^2_2 I\)
- **AOL:** Statement 1 with \(T_k = \text{diag}(\sum_{j=1}^n |W_k^T W_k|_{ij})\)
- **Control Theory (SLL):** \(T_k = \text{diag}(\sum_{j=1}^n |W_k^T W_k|_{ij} q_j / q_i)\).
Experimental Results

4 versions of SDP-based Lipchitz Network (SLL) (S, M, L, XL)

<table>
<thead>
<tr>
<th>Datasets</th>
<th>Models</th>
<th>Natural Accuracy</th>
<th>Provable Accuracy ((\varepsilon))</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td>36</td>
</tr>
<tr>
<td>CIFAR100</td>
<td>Cayley Large</td>
<td>43.3</td>
<td>29.2</td>
</tr>
<tr>
<td></td>
<td>SOC 20</td>
<td>48.3</td>
<td>34.4</td>
</tr>
<tr>
<td></td>
<td>SOC+ 20</td>
<td>47.8</td>
<td>34.8</td>
</tr>
<tr>
<td></td>
<td>CPL XL</td>
<td>47.8</td>
<td>33.4</td>
</tr>
<tr>
<td></td>
<td>AOL Large</td>
<td>43.7</td>
<td>33.7</td>
</tr>
<tr>
<td></td>
<td>SLL Small</td>
<td>45.8</td>
<td>34.7</td>
</tr>
<tr>
<td></td>
<td>SLL Medium</td>
<td>46.5</td>
<td>35.6</td>
</tr>
<tr>
<td></td>
<td>SLL Large</td>
<td>46.9</td>
<td>36.2</td>
</tr>
<tr>
<td></td>
<td>SLL X-Large</td>
<td>47.6</td>
<td>36.5</td>
</tr>
</tbody>
</table>

- Competitive results over CIFAR100 and TinyImageNet
- Many extensions: Lipschitz deep equilibrium models, neural ODEs, etc
Quadratic Constraints for Lipschitz Networks

- **Residual network**: \( x_{k+1} = x_k - G_k \sigma(W_k^T x_k + b_k) \) for \( k = 0, 1, \ldots, D \).

- **1-Lipschitz layer**: How to enforce \( \|x_{k+1} - x'_{k+1}\| \leq \|x_k - x'_k\| \)?

- \[ \begin{aligned} (x_{k+1} - x'_{k+1}) & = (x_k - x'_k) - G_k \left( \sigma(W_k^T x_k + b_k) - \sigma(W_k^T x'_k + b_k) \right) \\ \xi_{k+1} & \leq \xi_k - G_k w_k 
\end{aligned} \]

- We will use the property of \( \sigma \) to construct \( M_k \) such that we only need to look at the following set with \( A_k = I \) and \( B_k = -G_k \):

  \[
  \left\{ (\xi, w) : \xi_{k+1} = A_k \xi_k + B_k w_k, \begin{bmatrix} \xi_k \\ w_k \end{bmatrix}^T M_k \begin{bmatrix} \xi_k \\ w_k \end{bmatrix} \leq 0 \right\},
  \]

- Then we can ensure \( \|\xi_{k+1}\| \leq \|\xi_k\| \) via enforcing a SDP for the set:

  \[
  \begin{bmatrix} A_k^T P A_k - P \\ B_k^T P A_k \\ A_k^T P B_k \\ B_k^T P B_k \end{bmatrix} \preceq M_k \iff \begin{bmatrix} \xi_k \\ w_k \end{bmatrix}^T \begin{bmatrix} A_k A_k - I \\ B_k^T A_k \\ B_k^T B_k \end{bmatrix} \begin{bmatrix} \xi_k \\ w_k \end{bmatrix} \leq 0
  \]

  \[
  \|\xi_{k+1}\|^2 - \|\xi_k\|^2 = \|x_{k+1} - x'_{k+1}\|^2 - \|x_k - x'_k\|^2
  \]
Quadratic Constraints for Lipschitz Networks

• Since $\sigma$ is slope-restricted on $[0, 1]$, the following scalar-version incremental quadratic constraint holds with $m = 0$ and $L = 1$:

$$\begin{bmatrix} a - a' \\ \sigma(a) - \sigma(a') \end{bmatrix}^T \begin{bmatrix} 2mL & -(m + L) \\ -(m + L) & 2 \end{bmatrix} \begin{bmatrix} a - a' \\ \sigma(a) - \sigma(a') \end{bmatrix} \leq 0$$

$$\begin{bmatrix} 0 & -1 \\ -1 & 2 \end{bmatrix}$$

• The vector-version quadratic constraint: For diagonal $\Gamma_k \succeq 0$, we have

$$\begin{bmatrix} v_k - v'_k \\ \sigma(v_k) - \sigma(v'_k) \end{bmatrix}^T \begin{bmatrix} 0 & -\Gamma_k \\ -\Gamma_k & 2\Gamma_k \end{bmatrix} \begin{bmatrix} v_k - v'_k \\ \sigma(v_k) - \sigma(v'_k) \end{bmatrix} \leq 0$$

$$X_k$$

• Choosing $v_k = W_k^T x_k + b_k$ and $v'_k = W_k^T x'_k + b_k$, we have

$$\begin{bmatrix} W_k^T(x_k - x'_k) \\ \sigma(W_k^T x_k + b_k) - \sigma(W_k^T x'_k + b_k) \end{bmatrix}^T X_k \begin{bmatrix} W_k^T(x_k - x'_k) \\ \sigma(W_k^T x_k + b_k) - \sigma(W_k^T x'_k + b_k) \end{bmatrix} \leq 0$$
Quadratic Constraints for Lipschitz Networks

• We get
\[ \begin{bmatrix} \xi_k \\ w_k \end{bmatrix}^T M_k \begin{bmatrix} \xi_k \\ w_k \end{bmatrix} \leq 0 \]
with
\[ M_k = \begin{bmatrix} W_k & 0 \\ 0 & I \end{bmatrix} \begin{bmatrix} 0 & -\Gamma_k \\ -\Gamma_k & 2\Gamma_k \end{bmatrix} \begin{bmatrix} W_k^T & 0 \\ 0 & I \end{bmatrix} \begin{bmatrix} 0 \\ -\Gamma_k W_k^T \\ -\Gamma_k W_k^T & 2\Gamma_k \end{bmatrix} \]

Theorem

If there exists diagonal \( \Gamma_k \succeq 0 \) such that
\[ \begin{bmatrix} 0 & -G_k \\ -G_k^T & G_k^T G_k \end{bmatrix} \preceq \begin{bmatrix} 0 & -W_k \Gamma_k \\ -\Gamma_k W_k^T & 2\Gamma_k \end{bmatrix} \]
then the residual network \( x_{k+1} = x_k - G_k \sigma(W_k^T x_k + b_k) \) is 1-Lipschitz.

• Analytical solution: \( G_k = W_k \Gamma_k \) and \( \Gamma_k W_k^T W_k \Gamma_k \preceq 2\Gamma_k \).

• Suppose \( \Gamma_k \) is nonsingular, and \( T_k = 2\Gamma_k^{-1} \). Then the residual network \( x_{k+1} = x_k - 2W_k T_k^{-1} \sigma(W_k^T x_k + b_k) \) is 1-Lipschitz as long as \( T_k \succeq W_k^T W_k \).

Outline

• Control for Learning
  • Control Methods on Certifiably Robust Neural Networks
  • A Control Perspective on Stochastic Learning Algorithms

• Learning for Control
  • Global convergence of direct policy search on robust control
History: Computer-Assisted Proofs in Optimization

In the past ten years, much progress has been made in leveraging SDPs to assist the convergence rate analysis of optimization methods.

- Drori and Teboulle (MP2014): numerical worst-case bounds via the performance estimation problem (PEP) formulation
- Lessard, Recht, Packard (SIOPT2016): numerical linear rate bounds using integral quadratic constraints (IQC) from robust control theory
- Taylor, Hendrickx, Glineur (MP2017): interpolation conditions for PEPs
- H., Lessard (ICML2017): first SDP-based analytical proof for Nesterov’s accelerated rate
- H., Seiler, Ranzter (COLT2017): first paper on SDP-based convergence proofs for stochastic optimization using jump system theory and IQCs
- Van Scoy, Freeman, and Lynch (LCSS2017): first paper on control-oriented design of accelerated methods: triple momentum

Taken further by different groups

- inexact gradient methods, proximal gradient methods, conditional gradient methods, operator splitting methods, mirror descent methods, distributed gradient methods, monotone inclusion problems
Stochastic Methods for Machine Learning

- Many learning tasks (regression/classification) lead to finite-sum ERM

\[
\min_{x \in \mathbb{R}^p} \frac{1}{n} \sum_{i=1}^{n} f_i(x)
\]

where \( f_i(x) = l_i(x) + \lambda R(x) \) (\( l_i \) is the loss, and \( R \) avoids over-fitting).

- Stochastic gradient descent (SGD):

\[
x_{k+1} = x_k - \alpha \nabla f_{i_k}(x_k)
\]

- Inexact oracle:

\[
x_{k+1} = x_k - \alpha (\nabla f_{i_k}(x_k) + e_k) \text{ where } \| e_k \| \leq \delta \| \nabla f_{i_k}(x_k) \| \text{ (the angle } \theta \text{ between } (e_k + \nabla f_{i_k}(x_k)) \text{ and } \nabla f_{i_k}(x_k) \text{ satisfies } | \sin(\theta) | \leq \delta)
\]

- Algorithm change: SAG (SRF2017) vs. SAGA (DBL2014)

\[
\text{SAG: } x_{k+1}^k = x_k^k - \alpha \left( \frac{\nabla f_{i_k}(x_k) - y_i^{k_k}}{n} + \frac{1}{n} \sum_{i=1}^{n} y_i^k \right)
\]

\[
\text{SAGA: } x_{k+1}^k = x_k^k - \alpha \left( \nabla f_{i_k}(x_k) - y_i^{k_k} + \frac{1}{n} \sum_{i=1}^{n} y_i^k \right)
\]

where \( y_i^{k+1} := \left\{ \begin{array}{ll}
\nabla f_i(x^k) & \text{if } i = i_k \\
y_i^k & \text{otherwise}
\end{array} \right. \)

- Markov assumption: In reinforcement learning, \( \{i_k\} \) can be Markovian
My Focus: Unified Analysis of Stochastic Methods

Assumption

• $f_i$ smooth, $f$ RSI
• $i_k$ is IID or Markovian
• Oracle is exact or inexact
• many other possibilities

Method

• SGD
• SAGA-like methods
• Temporal difference learning

Bound

• $\mathbb{E}\|x_k - x^*\|^2 \leq c_2 \rho^k + O(\alpha)$
• $\mathbb{E}\|x_k - x^*\|^2 \leq c_2 \rho^k$
• Other forms

How to automate rate analysis of stochastic learning algorithms? Use numerical semidefinite programs to support search for analytical proofs?

assumption + method $\implies$ bound
My Focus: Stochastic Methods for Learning

In the deterministic setting, we just need to show that the trajectories generated by optimization methods belong to the following set:

\[
\{(\xi, w, v) : \xi_{k+1} = A\xi_k + Bw_k, \ v_k = C\xi_k, \ \begin{bmatrix} v_k \\ w_k \end{bmatrix}^T M_j \begin{bmatrix} v_k \\ w_k \end{bmatrix} \leq \Lambda_j, j \in \Pi \}\n\]

What to do for stochastic optimization (e.g. \(x_{k+1} = x_k - \alpha \nabla f_i(x_k)\) where \(i_k \in \{1, \cdots, n\}\) is sampled)?

- **Stochastic quadratic constraints**: Show that the trajectories generated by stochastic optimization methods belong to the following set:

\[
\{(\xi, w, v) : \xi_{k+1} = A\xi_k + Bw_k, \ v_k = C\xi_k, \mathbb{E} \begin{bmatrix} v_k \\ w_k \end{bmatrix}^T M_j \begin{bmatrix} v_k \\ w_k \end{bmatrix} \leq \Lambda_j, j \in \Pi \}\n\]

- **Jump system approach**: Show that the trajectories generated by stochastic optimization methods belong to the following set:

\[
\{(\xi, w, v) : \xi_{k+1} = A_i\xi_k + B_iw_k, \ v_k = C_i\xi_k, \ \begin{bmatrix} v_k \\ w_k \end{bmatrix}^T M_j \begin{bmatrix} v_k \\ w_k \end{bmatrix} \leq \Lambda_j, j \in \Pi \}\n\]

where \(A_{i_k} \in \{A_1, \cdots, A_n\}\), \(B_{i_k} \in \{B_1, \cdots, B_n\}\), and \(C_{i_k} \in \{C_1, \cdots, C_n\}\)
Stochastic Quadratic Constraints

Suppose we can show that the trajectories generated by stochastic optimization methods belong to the following set:

\[
\left\{ (\xi, w, v) : \xi_{k+1} = A\xi_k + Bw_k, \ v_k = C\xi_k, \ \mathbb{E} \begin{bmatrix} v_k \\ w_k \end{bmatrix}^T M_j \begin{bmatrix} v_k \\ w_k \end{bmatrix} \leq \Lambda_j, j \in \Pi \right\}
\]

Theorem

If there exists a positive definite matrix \( P \), non-negative \( \lambda_j \) and \( 0 < \rho < 1 \) s.t.

\[
\begin{bmatrix}
A^T PA - \rho^2 P & A^T PB \\
B^T PA & B^T PB
\end{bmatrix} \preceq \sum_{j \in \Pi} \lambda_j \begin{bmatrix} C & 0 \\ 0 & I \end{bmatrix}^T M_j \begin{bmatrix} C & 0 \\ 0 & I \end{bmatrix}
\]

then \( \mathbb{E} \xi_{k+1}^T P \xi_{k+1} \leq \rho^2 \mathbb{E} \xi_k^T P \xi_k + \sum_{j \in \Pi} \lambda_j \Lambda_j \).

\[
\begin{bmatrix}
\xi_k \\
w_k
\end{bmatrix}^T \begin{bmatrix}
A^T PA - \rho^2 P & A^T PB \\
B^T PA & B^T PB
\end{bmatrix} \begin{bmatrix}
\xi_k \\
w_k
\end{bmatrix} \leq \sum_{j \in \Pi} \lambda_j \begin{bmatrix} v_k \\ w_k \end{bmatrix}^T M_j \begin{bmatrix} v_k \\ w_k \end{bmatrix}
\]

\( \xi_{k+1}^T P \xi_{k+1} - \rho^2 \xi_k^T P \xi_k \)

Then take expectation and apply the expected quadratic constraints!
Main Result: Analysis of Biased SGD

- Consider \( x_{k+1} = x_k - \alpha (\nabla f_{i_k}(x_k) + e_k) \) with \( \|e_k\|^2 \leq \delta^2 \|\nabla f_{i_k}(x_k)\|^2 + c^2 \)

- If \( c = 0 \), the bound means the angle \( \theta \) between \( (e_k + \nabla f_{i_k}(x_k)) \) and \( \nabla f_{i_k}(x_k) \) satisfies \( |\sin(\theta)| \leq \delta \)

- Rewritten as \( (x_{k+1} - x^*) = (x_k - x^*) + [-\alpha I \quad -\alpha I] \begin{bmatrix} \nabla f_{i_k}(x_k) \\ e_k \end{bmatrix} \)

- Assume the restricted secant inequality \( \nabla f(x)^T (x - x^*) \geq m \|x - x^*\|^2 \)

- Assume \( f_i \) is \( L \)-smooth, i.e. \( \|\nabla f_i(x) - \nabla f_i(x^*)\| \leq L \|x - x^*\|\)

1st QC: \( \mathbb{E} \begin{bmatrix} x_k - x^* \\ \nabla f_{i_k}(x_k) \\ e_k \end{bmatrix}^T \begin{bmatrix} 2mI & -I & 0 \\ -I & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} x_k - x^* \\ \nabla f_{i_k}(x_k) \\ e_k \end{bmatrix} \leq 0 \begin{bmatrix} \Lambda_1 \end{bmatrix} \)

2nd QC: \( \mathbb{E} \begin{bmatrix} x_k - x^* \\ \nabla f_{i_k}(x_k) \\ e_k \end{bmatrix}^T \begin{bmatrix} -2L^2I & 0 & 0 \\ 0 & I & 0 \\ 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} x_k - x^* \\ \nabla f_{i_k}(x_k) \\ e_k \end{bmatrix} \leq 0 \begin{bmatrix} \Lambda_2 \end{bmatrix} \)

\( \Lambda_1 = \frac{2}{n} \sum_{i=1}^{n} \|\nabla f_i(x^*)\|^2 \)
Main Result: Analysis of Biased SGD

• We can rewrite \( \|e_k\|^2 \leq \delta^2 \|\nabla f_{i_k}(x_k)\|^2 + c^2 \) as

\[
\mathbb{E} \begin{bmatrix} x_k - x^* \\ \nabla f_{i_k}(x_k) \\ e_k \end{bmatrix}^T \begin{bmatrix} 0 & 0 & 0 \\ 0 & -\delta^2 I & 0 \\ 0 & 0 & I \end{bmatrix} \begin{bmatrix} x_k - x^* \\ \nabla f_{i_k}(x_k) \\ e_k \end{bmatrix} \leq \frac{c^2}{\Lambda_3} \]

\[ M_3 \]

• We have \( A = I \), \( B = \begin{bmatrix} -\alpha I & -\alpha I \end{bmatrix} \), \( C = I \), and the following SDP

\[
\begin{bmatrix} A^T PA - \rho^2 P & A^T PB \\ B^T PA & B^T PB \end{bmatrix} \preceq \sum_{j=1}^3 \lambda_j \begin{bmatrix} C & 0 \\ 0 & I \end{bmatrix}^T M_j \begin{bmatrix} C & 0 \\ 0 & I \end{bmatrix}
\]

• Biased SGD satisfies \( \mathbb{E}\|x_{k+1} - x^*\|^2 \leq \rho^2 \mathbb{E}\|x_k - x^*\|^2 + \lambda_2 \Lambda_2 + \lambda_3 c^2 \) if

\[
\begin{bmatrix} 1 - \rho^2 & -\alpha & -\alpha \\ -\alpha & \alpha^2 - \delta^2 \lambda_3 & \alpha^2 \\ -\alpha & \alpha^2 & \alpha^2 + \lambda_3 \end{bmatrix} + \lambda_1 \begin{bmatrix} -2m & 1 & 0 \\ 1 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix} + \lambda_2 \begin{bmatrix} 2L^2 & 0 & 0 \\ 0 & -1 & 0 \\ 0 & 0 & 0 \end{bmatrix} \preceq 0
\]
Main Result: Analysis of Biased SGD

- Given $\mathbb{E}\|x_0 - x^*\|^2 \leq U_0$, set $U_{k+1} = \min(\rho^2 U_k + \lambda_2 \Lambda_2 + \lambda_3 c^2)$ with

$$\begin{bmatrix}
1 - \rho^2 & -\alpha & -\alpha \\
-\alpha & \alpha^2 - \delta^2\Lambda_3 & \alpha^2 \\
-\alpha & \alpha^2 & \alpha^2 + \lambda_3
\end{bmatrix} + \lambda_1 \begin{bmatrix}
-2m & 1 & 0 \\
1 & 0 & 0 \\
0 & 0 & 0
\end{bmatrix} + \lambda_2 \begin{bmatrix}
2L^2 & 0 & 0 \\
0 & -1 & 0 \\
0 & 0 & 0
\end{bmatrix} \preceq 0$$

then we have $\mathbb{E}\|x_k - x^*\|^2 \leq U_k$. This leads to a sequential SDP problem.

- This problem has an exact solution

$$U_{k+1} = \left(\alpha \sqrt{c^2 + \delta^2 \Lambda_2 + 2L^2 \delta^2 U_k} + \sqrt{(1 - 2m\alpha + 2L^2 \alpha^2)U_k + \Lambda_2 \alpha^2}\right)^2$$

- $\lim_{k \to \infty} U_k = \frac{c^2 + \delta^2 \Lambda_2}{m^2 - 2\delta^2 L^2} + \frac{m(c^2(2L^2 - m^2) + (1 - \delta^2)\Lambda_2 m^2)}{(m^2 - 2\delta^2 L^2)^2} \alpha + O(\alpha^2)$

- Rate $= 1 - \frac{m^2 - 2\delta^2 L^2}{m} \alpha + O(\alpha^2)$

- For different assumptions, modify $(M_j, \Lambda_j)$!


- Syed, Dall'Anese, H.. Bounds for the tracking error and dynamic regret of inexact online optimization methods: A unified analysis via sequential SDPs.
Jump System Approach

\[
\frac{1}{n} \sum_{i=1}^{n} \left[ A_i^T P A_i - \rho^2 P \begin{bmatrix} B_i^T P A_i & B_i^T P B_i \end{bmatrix} \right] \preceq \sum_{j \in \Pi} \lambda_j \begin{bmatrix} C & 0 \\ 0 & I \end{bmatrix}^T M_j \begin{bmatrix} C & 0 \\ 0 & I \end{bmatrix}
\]

Pros:

- General enough to handle many algorithms: H., Seiler, Rantzer (COLT2017)

<table>
<thead>
<tr>
<th>Method</th>
<th>( \tilde{A}_{ik} )</th>
<th>( \tilde{B}_{ik} )</th>
<th>( \tilde{C} )</th>
</tr>
</thead>
</table>
| SAGA   | \[
\begin{bmatrix}
I_n - e_{ik} e_{ik}^T \\
-\frac{\alpha}{n} (e - ne_{ik})^T
\end{bmatrix} \tilde{0}
\] | \[
\begin{bmatrix}
e_{ik} e_{ik}^T \\
-\alpha e_{ik}^T
\end{bmatrix}
\] | \[
\tilde{0}^T \\
1
\]
| SAG    | \[
\begin{bmatrix}
I_n - e_{ik} e_{ik}^T \\
-\frac{\alpha}{n} (e - e_{ik})^T
\end{bmatrix} \tilde{0}
\] | \[
\begin{bmatrix}
e_{ik} e_{ik}^T \\
-\alpha e_{ik}^T
\end{bmatrix}
\] | \[
\tilde{0}^T \\
1
\]

- General enough to handle Markov \( \{i_k\} \): Syed and H. (NeurIPS2019), Guo and H. (ACC2022a,2022b)

Cons:

- SDPs are much bigger than the ones obtained from stochastic quadratic constraints, and we have to exploit SDP structures for simplifications
Control for Learning: Summary

- Iterative learning algorithms and neural network layers can be thought as feedback control systems.

- The quadratic constraint approach from control theory can be leveraged to formulate SDP conditions for machine learning research.

- Different from the study in control, now we want to obtain analytical solutions of the SDPs!
Outline

• Control for Learning
  • Control methods on certifiably robust neural networks
  • A control perspective on stochastic learning algorithms

• Learning for Control
  • Global convergence of direct policy search on robust control