Tour Guide: Deep Learning in Trajectory Optimization

Jay Guthrie
jguthr@stanford.edu

Summary

- Real-time trajectory optimization for autonomous vehicles requires solving optimization problems **fast**
- Solver runtime can be reduced by providing a good initial guess to the solution ("warm-start')
- We train a fully-connected neural network on a family of optimal trajectories for a reusable launch vehicle (RLV)
- We achieve a 2.2x faster solver by warm-starting with our neural network

Related Work

- Using deep learning to "optimize the optimizers" is a relatively new research thread
- (Mansard, 2018) trains a neural network offline to provide warm-start solutions for a UAV trajectory planner
- RLV problem is higher dimension, longer time horizon, and more challenging dynamics

Data/Features/Models

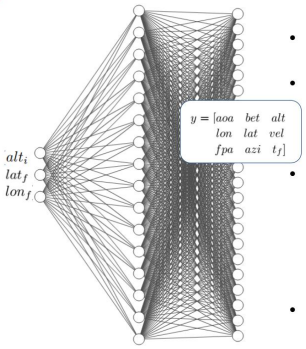
- 14,000 optimal trajectories for how the RLV should travel from initial altitude to a final latitude & longitude

$$x = [alt_i \quad lat_f \quad lon_f] \in \mathbb{R}^3$$
$$y = [aoa \quad bet \quad alt \quad lon \quad lat \quad vel \quad fpa \quad azi \quad t_f] \in \mathbb{R}^{801}$$

- Generated with GPOPS-II optimal control software (12 hours)
- Data shows clear trends → Find a low-dimensional representation for easy implementation on flight computer
- Multivariate regression problem. Penalize mean squared error

$$J(y, \hat{y}) = \frac{1}{m}\frac{1}{n_y}\sum_{i=1}^{m}\sum_{j=1}^{n_y}(y_j - \hat{y}_j)^2$$

- Used 90/10 train/test split. Data shifted and scaled on both input and output (to ensure equal weighting in cost function)
- TensorFlow + Keras for neural network training using ADAM



Results & Discussion

- To ensure small computation footprint, searched for minimum number of hidden units necessary to achieve < 1% error on train/test set
- Final architecture: Single hidden layer with 16 ReLUs (very small!)

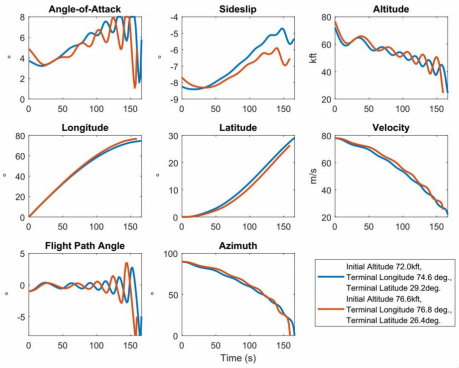
	Min. Loss	Max. Loss	Mean Loss
Train	0.0007	0.018	0.0077
Test	0.0007	0.016	0.0071

- Utilized neural network to warm-start solver on 1000 previously unseen problems → 2.2x faster than cold-start

	Minimum (s)	Maximum (s)	Mean (s)
Cold-Start	0.64	8.08	1.76
Warm-Start (w/Neural Network)	0.25	1.31	0.77
Δ = Cold Start - Warm Start	0.09	6.96	1.00

- Promising result. Future work will build upon this framework to tackle higher dimensional problems.

Example Data



References

[1] M. Zhong, M. Johnson, Y. Tassa, T. Erez, and E. Todorov, "Value Function Approximation and Model Predictive Control," 2013 IEEE Symposium on Adaptive Dynamic Programming and Reinforcement Learning pp.100-107

[2] W. Merkt, W. Ivan, V. Vijayakumar, "Leveraging Precomputation with Problem Encoding for Warm-Starting Trajectory Optimization in Complex Environments," 2018 IEEE/RSJ International Conference on Intelligent Robots and Systems. pp. 5877-5884

[3] N. Mansard, A. DelPrete, M. Geisert, S. Tonneau and O. Stasse, "Using a Memory of Motion to Efficiently Warm-Start a Nonlinear Predictive Controller," 2018 IEEE International Conference on Robotics and Automation (ICRA), Brisbane, QLD, 2018, pp. 2986-2993.

[4] J. Betts Practical Methods for Optimal Control and Estimation Using Nonlinear Programming.SIAM Press, Philadelphia, PA. 2010

[5] M. Patterson, A. Rao "GPOPS-II A MATLAB Software for Solving Multiple-Phase Optimal Control Problems using hp-Adaptive Gaussian Quadrature Collocation Methods and Sparse Nonlinear Programming," ACM Transactions on Mathematical Software, Vol. 41, No. 1, October 2014.

[6] F. Chollet et. al, "Keras", 2015. https://keras.io

[7] M. Abadi et. al, "TensorFlow: Large-Scale Machine Learning on Heterogeneous Systems," 2015. https://www.tensorflow.org

[8] A. Wachter, L. Biegler, "On the Implementation of a Primal-Dual Interior Point Filter Line Search Algorithm for Large-Scale Nonlinear Programming," Mathematical Programming 106(1), pp. 25-27, 2006

[9] D. Kingma, "Adam: A Method for Stochastic Optimization," Proceedings of the 3rd International Conference on Learning Representations, 2014.

[10] F. Pedregosa et. al, "Scikit-learn: Machine Learning in Python", Journal of Machine Learning Research, 2011

Presentation video & code available at

https://github.com/guthriejd1/cs230_project