Predict Globally, Correct Locally: Parallel-in-Time Optimal Control of Neural Networks

Panos Parpas

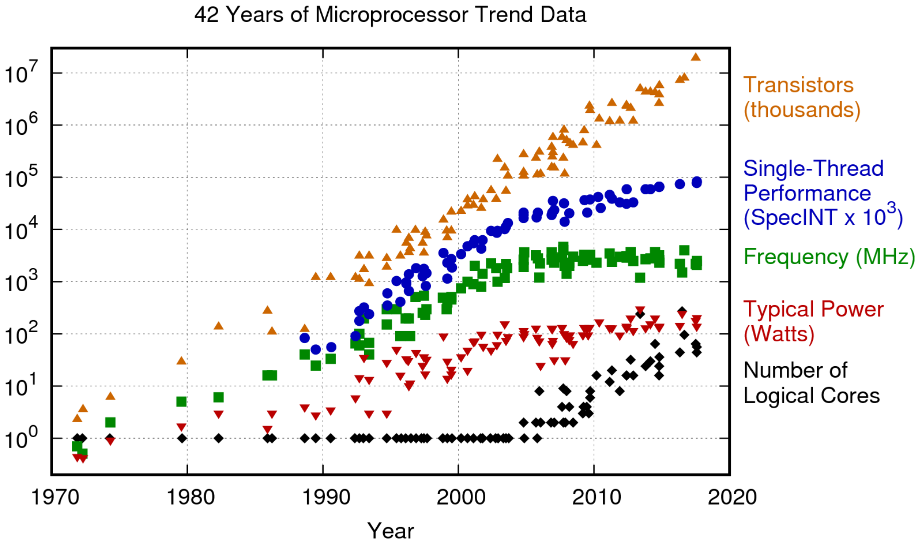
Department of Computing
Imperial College London
www.doc.ic.ac.uk/~pp500
p.parpas@imperial.ac.uk

13-Feb-2019

Symposium on Machine Learning and Dynamical Systems
Imperial College London, Feb, 2019.

P.P, C. Muir. Predict Globally, Correct Locally: Parallel-in-Time Optimal Control of Neural Networks, https://arxiv.org/pdf/1902.02542v1.pdf

Microprocessor Trends → Distributed Computation



Original data up to the year 2010 collected and plotted by M. Horowitz, F. Labonte, O. Shacham, K. Olukotun, L. Hammond, and C. Batten
New plot and data collected for 2010-2017 by K. Rupp

Memory Energy Costs for Memory Access

40nm, 8-core processor with an 8MB last-level cache†

Integer	
Add	
8 bit	0.03pJ
32 bit	0.1pJ
Mult	
8 bit	0.2pJ
32 bit	3.1pJ

FP	
FAdd	
16 bit	0.4pJ
32 bit	0.9pJ
FMult	
16 bit	1.1pJ
32 bit	3.7pJ

Memory	
Cache	(64bit)
8KB	10pJ
32KB	20pJ
1MB	100pJ
DRAM	1.3-2.6nJ

† M. Horowitz, *ComputingâĂŹs Energy Problem (and what we can do about it)*, ISSCC 2014

Memory Energy Costs for Memory Access

40nm, 8-core processor with an 8MB last-level cache†

Integer	
Add	
8 bit	0.03pJ
32 bit	0.1pJ
Mult	
8 bit	0.2pJ
32 bit	3.1pJ

FP	
FAdd	
16 bit	0.4pJ
32 bit	0.9pJ
FMult	
16 bit	1.1pJ
32 bit	3.7pJ

Memory	
Cache	(64bit)
8KB	10pJ
32KB	20pJ
1MB	100pJ
DRAM	1.3-2.6nJ

† M. Horowitz, *ComputingâĂŹs Energy Problem (and what we can do about it)*, ISSCC 2014

Fast and Energy Efficient Distributed Computation:

- Decompose problem in small "sub-spaces"
- Avoid communication
- Use more FP operations

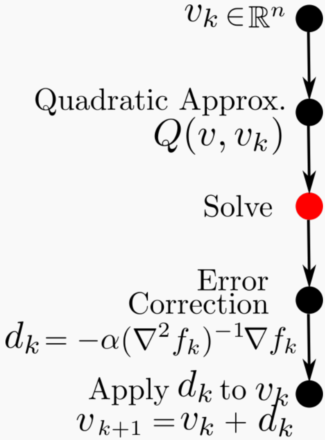
Multi-level/resolution Algorithms

$$\boxed{\min_{v \in \mathbb{R}^n} f(v)}$$

Multi-level/resolution Algorithms

$$Q(v, v_k) = v(x_k) + \langle \nabla v_k, v - v_k \rangle + \frac{1}{2\alpha_k} \langle v - v_k, \nabla^2 f_k (v - v_k) \rangle$$

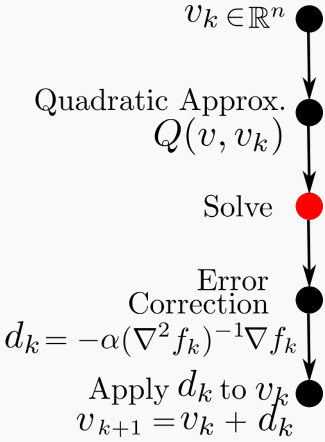
Quadratic Approximation



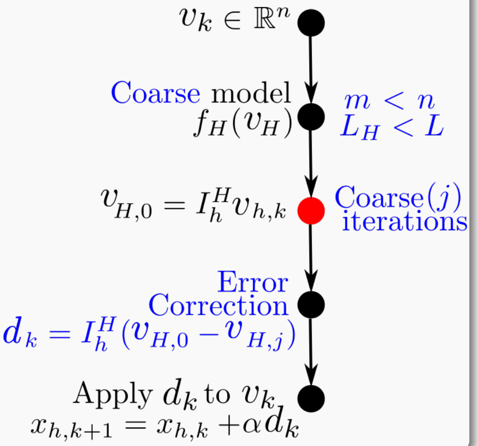
Multi-level/resolution Algorithms

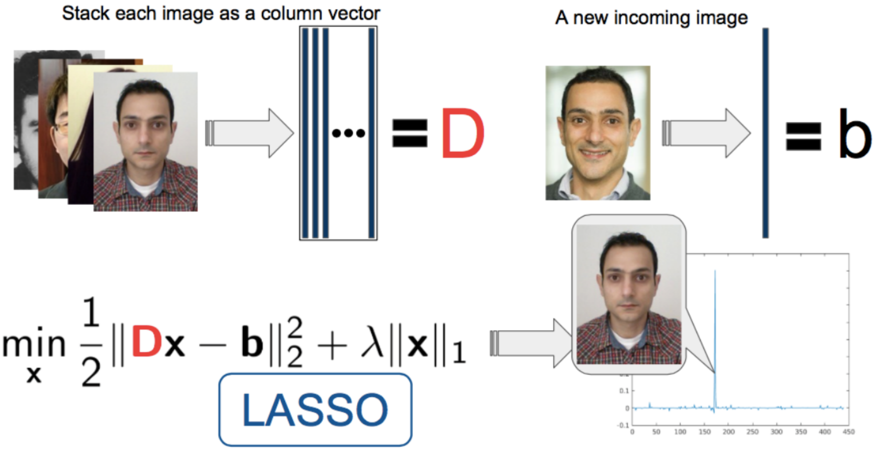
Use a low resolution problem with *favorable characteristics*

Quadratic Approximation



Coarse Approximation





V. Hovhannisyan, P.P, and S. Zafeiriou. *MAGMA: Multi-level accelerated gradient mirror descent algorithm for large-scale convex composite minimization*, SIAM J. on Imag. Sci., 2016.

P.P. *A Multilevel Proximal Gradient Algorithm for Large Scale Optimization*, SIAM Journal on Scientific Computing, Vol. 39, Issue 5, Nov. 2017.

V. Hovhannisyan,Y. Panagakis,P.P, S. Zafeiriou *Fast Multilevel Algorithms for Compressive Principle Component Pursuit. SIAM Journal on Imaging Sciences 2019.*

Distributed Optimization Algorithms

$$\boxed{v^\star \in \arg\min_{v \in \mathbb{R}^n} f(v_1, v_1, \ldots, v_n)}$$

- **Coordinate methods**: Processor (i) updates coordinate i

$$v_i \leftarrow v_i + d_i$$

- **Duality methods**: Copy model, enforce consensus via penalties.

Properties:

- (A/As)ynchronous variants.
- Slow (sub-linear)
- Not optimized for communication/energy
- Sensitive to parameter choice (duality methods)
- Randomized variants (e.g. SGD) are hard to parallelize

$$Y = F(X)$$

Supervised Learning: Learn F given $\{Y_i, X_i\}_{i=1}^{M}$:

- Additive approximation: $F_\phi(X) = \sum_{i=1}^{K} c_i \phi(X)$ (e.g. SVMs)

Neural Networks & Dynamical Systems

$$Y = F(X)$$

Supervised Learning: Learn F given $\{Y_i, X_i\}_{i=1}^{M}$:

- Additive approximation: $F_\phi(X) = \sum_{i=1}^{K} c_i \phi(X)$ (e.g. SVMs)
- Approximation by composition: $F_u(X) = F_w^L(\dots F_u^1(F_u^0(X)))$ (Neural Networks) e.g.

$$F_u(X) = \tanh(A^\top X + b), U = [A, b]$$

Neural Networks & Dynamical Systems

$$Y = F(X)$$

Supervised Learning: Learn F given $\{Y_i, X_i\}_{i=1}^{M}$:

- Additive approximation: $F_\phi(X) = \sum_{i=1}^{K} c_i \phi(X)$ (e.g. SVMs)
- Approximation by composition: $F_u(X) = F_w^L(\dots F_u^1(F_u^0(X)))$ (Neural Networks) e.g.

$$F_u(X) = \tanh(A^\top X + b), U = [A, b]$$

$$\min_{U(t)} \sum_{i=1}^{n} l(X_i(T), Y_i(T)) + \sum_{t=0}^{T} r(U(t))$$
$$X_i(t+1) = F(X_i(t), U(t), t), \quad X_i(0) = x_{0,i}$$

The Dynamical Systems View

$$\min_{W(t)} \sum_{i=1}^{n} l(X_i(T), Y_i(T)) + \int_0^T r(W(t))dt$$

$$\frac{dX_i(t)}{dt} = F(X_i(t), W(t), t), \quad X_i(0) = x_{0,i}$$

The discretized system may not be stable:

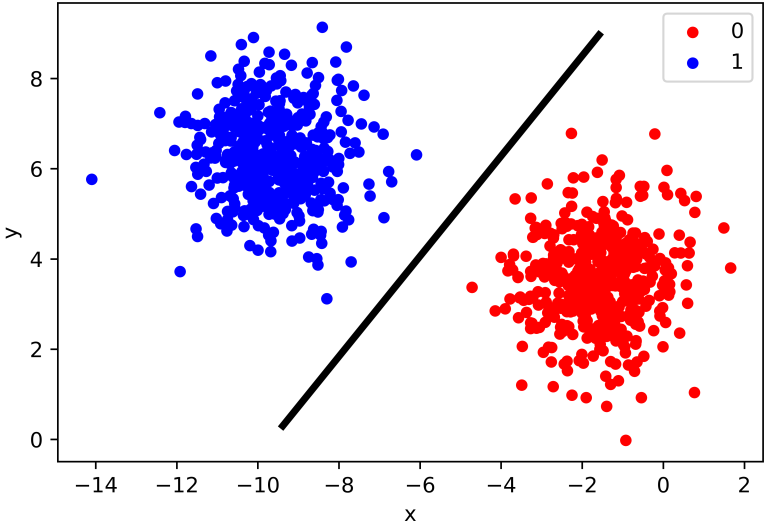
- Solutions diverge to infinity/zero (exploding/vanishing gradients)
- Small perturbations can fool the classifier (adversarial attacks)

Benefits:

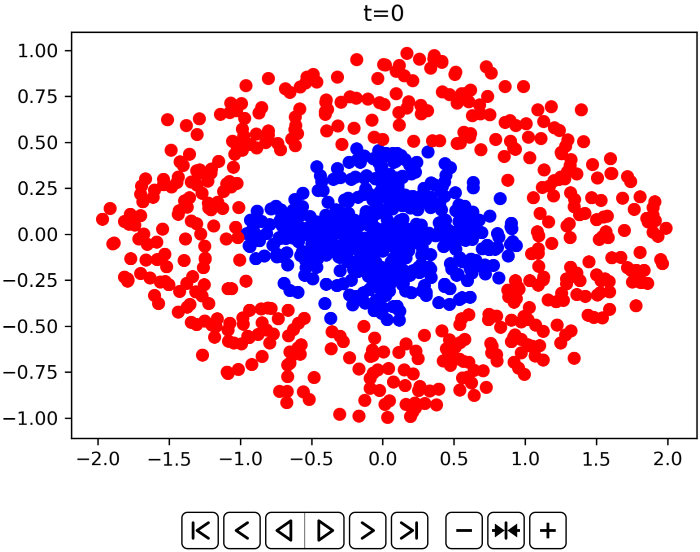
- Train with less data & hyper-parameters
- Rigorous mathematical framework to understand generalization

E.Haber, L.Ruthotto *Stable Architectures for Deep Neural Networks.* Inverse Problems, 2017

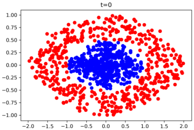
Li, Q., Chen, L., Tai, C., & Weinan, E. Maximum principle based algorithms for deep learning. JMLR, 2017.


0
1
y
x
8
6
4
2
0
−14
−12
−10
−8
−6
−4
−2
0
2

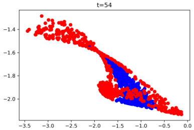
Neural Networks & Dynamical Systems



Serial-in-Time Optimization



$\rightsquigarrow$



Algorithm 1: $\texttt{Forward}(\delta, X_s, t_0, t_1, \{U(t)\}_{t=t_0}^{t=t_1})$

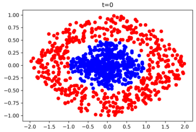
$t \leftarrow t_0,\ \ X(t) = X_s$

while$(t \leq t_1)$ **do**

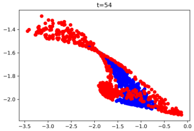
$\quad X(t+\delta) = f_t^{\delta}(X(t), U(t)),\ \ t \leftarrow t + \delta$

return $X(t), t_0 \leq t \leq t_1$

Serial-in-Time Optimization



$P(T, T-1, \ldots, 0)$



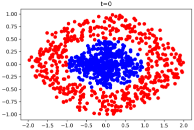
Algorithm 2: `Backward`$(\delta, P_e, t_0, t_1, \{U(t), X(t)\}_{t=t_0}^{t=t_1})$

$t \leftarrow t_1,\ P(t_1) = P_e$

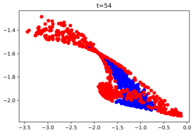
while$(t \geq t_0)$ **do**

$\quad P(t-\delta) = -\langle \nabla_x f_t^{\delta}(X(t), U(t), P(t) \rangle,\ t \leftarrow t - \delta$

return $P(t), t_0 \leq t \leq t_1$



$P(T, T-1, \dots, 0)$



Algorithm 3: `Serial-in-time Update Control`

Let $X^k(0)$ be a random sample from $[X]$.

$X^k(t) =$ `Forward`$(\delta, X^k(0), 0, T_\delta, U^k(t)), 0 \leq t \leq T_\delta$

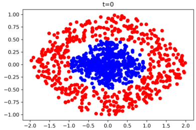
$P^k(T_\delta) = \nabla_x \Phi(X^k(T_\delta))$

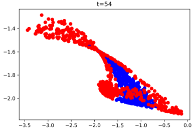
$P^k(t)=$`Backward`$(\delta, P^k(T), 0, T, U^k(t)), 0 \leq t \leq T_\delta$

`Update control` for $0 \leq t \leq T_\delta - 1$

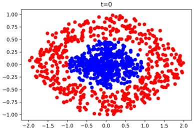
$$U^{k+1}(t) = \mathcal{A}(U^k(t), X^k(t), P^k(t+\delta)). \quad (1)$$

Global Prediction Phase: Predict Initial Conditions

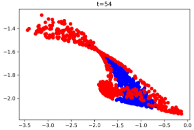




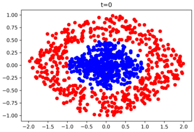
Global Prediction Phase: Predict Initial Conditions

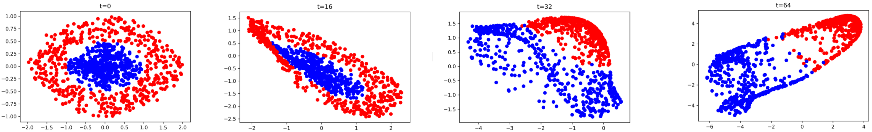


$P(T, T-1, \ldots, 0)$

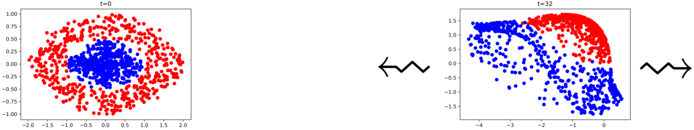


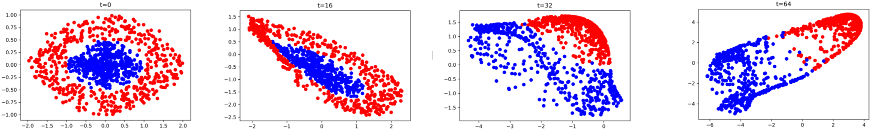
Global Prediction Phase: Predict Initial Conditions



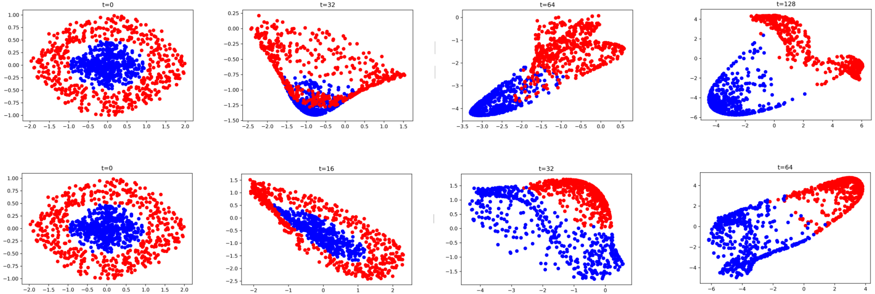


Global Prediction Phase: Predict Initial Conditions

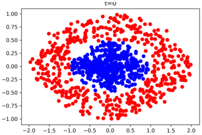


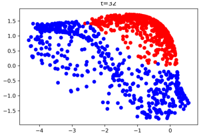


Global Prediction Phase: Predict Initial Conditions



Local Correction Phase





Algorithm 4: Parallel-in-Time Optimization

Global Prediction: Compute $\{U_t^0, P_t^0, X_t^0]\}_{t=0}^T$.

Processor A	Processor B
Backward solve: $\widehat{P}_s^k = \mathcal{L}[\mathcal{I}_k]$ $\texttt{Backward}(\delta, \widehat{P}_s^k, 0, s, U^k)$ **Update:** $U_t^{k+1} = \mathcal{A}(X_t^k, \widehat{P}_t^k)$ **Forward solve:** $\texttt{Forward}(\delta, X_0^k, 0, s, U_t^k)$ **Synchronization:** Send X_s^k to Processor B.	**Forward solve:** $\texttt{Forward}(\delta, X_s^k, s, T_\delta, U_t^k)$ **Backward solve:** $P_{T_\delta}^k = \nabla_x(X_{T_\delta}^k)$ $\texttt{Backward}(P_{T_\delta}^k, T_\delta, s, U^k)$ **Update:** $U_t^{k+1} = \mathcal{A}(X_t^k, P_t^k)$ **Synchronization:** Send P_s^k to Processor A.

Local Correction Phase

- $\mathcal{I}_k$: observed state/co-state pairs
- Global prediction phase: $\mathcal{I}_0 = \{(X^i(s), P^i(s)), i = 0, \ldots, H-1\}$.

$$\min_{A,B} L[\mathcal{I}_k] = \sum_{(X^i(s), P^i(s)) \in \mathcal{I}_0} \|AX_i(s) + B - P_i(X_i(s))\|^2.$$

$$\hat{P}(s) \approx A^\star X(s) + B^\star$$

Convergence Analysis

Theorem

Step-size satisfies the following conditions,

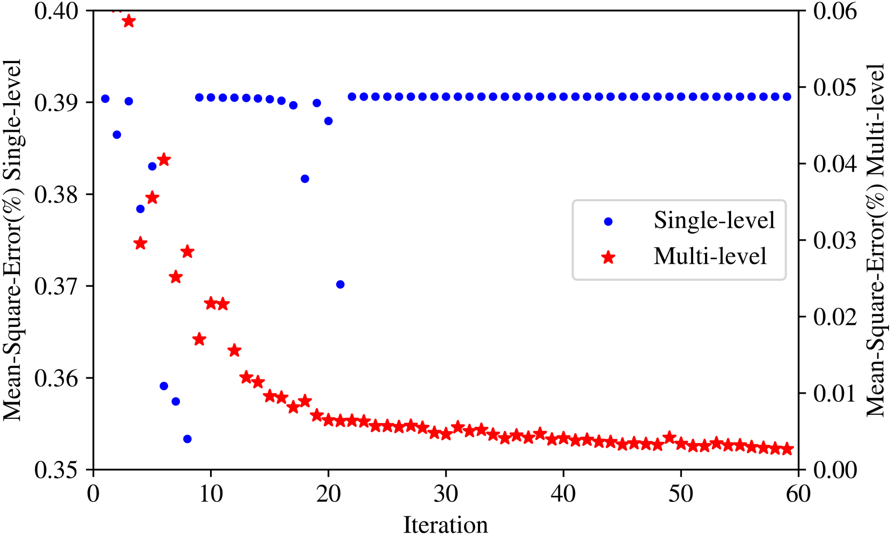
$$\sum_{k=1}^{\infty} \eta_k = \infty, \ \sum_{k=1}^{\infty} \eta_k^2 < \infty.$$

Approximation error in co-state:

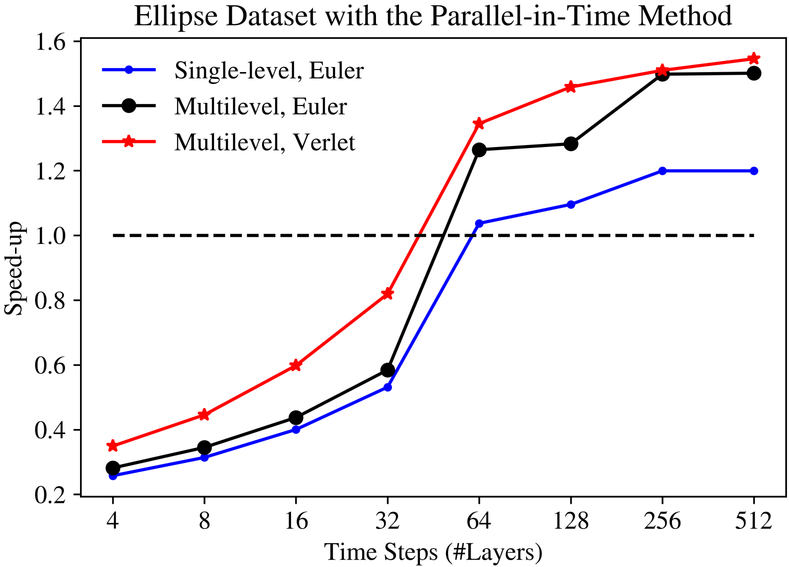
$$\|\widehat{P}^{\delta}(t) - P^{\delta}(t)\| \leq \epsilon_p \eta \|\widehat{P}^{\delta}(t)\|.$$

Then, $\lim_{M\to\infty} \frac{1}{H_M}\mathbb{E}\left(\sum_{k=1}^{M} \eta_k \|\nabla J(U^k)\|^2\right) = 0,$ *where* $H_M = \sum_{k=1}^{M} \eta_k.$

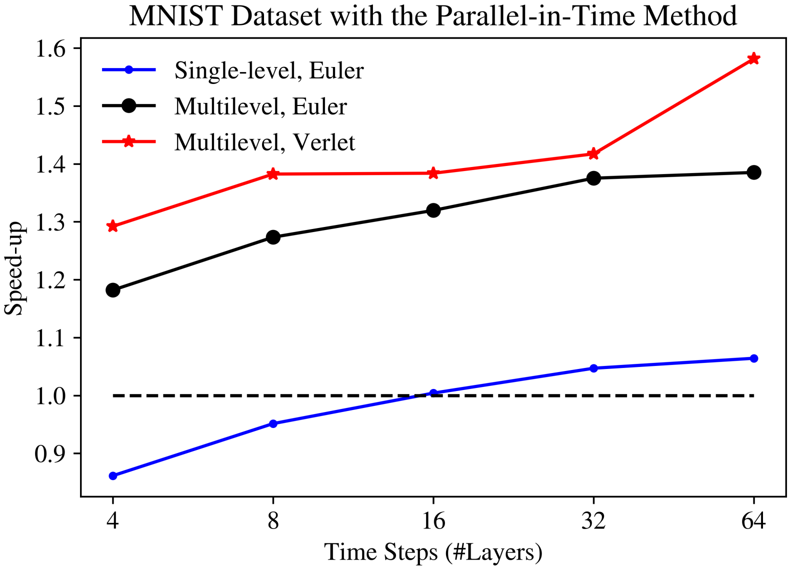
Improved Stability with Global Prediction Phase



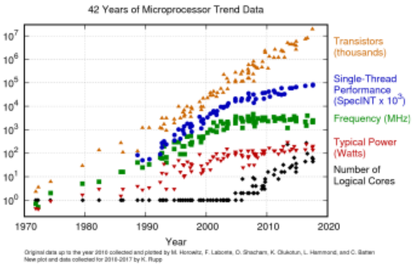
Results – Ellipse Model

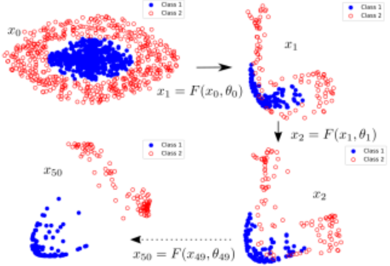


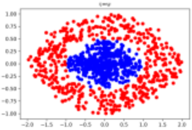
Results – MNIST Model

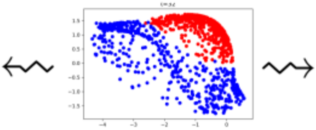


Conclusions









P.P, C. Muir. *Predict Globally, Correct Locally: Parallel-in-Time Optimal Control of Neural Networks*, `https://arxiv.org/pdf/1902.02542v1.pdf`