DDEKF Learning for Fast Nonlinear Adaptive Inverse Control

Dr. Gregory L. Plett and Hans Böttrich

Electrical and Computer Engineering Department
University of Colorado at Colorado Springs
P.O. Box 7150, Colorado Springs, CO 80933–7150
Overview

- Dynamic NN filtering:
  - Structure, RTRL, DDEKF.
- Nonlinear AIC:
  - System identification;
  - Adaptive feedforward control;
  - Adaptive disturbance cancellation;
  - BPTM-RTRL, BPTM-DDEKF.
- Illustrative example.
\[ y_k = f(x_k, x_{k-1}, \ldots, x_{k-n}, y_{k-1}, y_{k-2}, \ldots y_{k-m}, W). \]
Real Time Recurrent Learning (RTRL)

- Use a learning cost function $J_k = \frac{1}{2} e_k^T e_k$.
- Then, using first-order gradient descent,

$$\Delta W = -\eta \frac{dJ_k}{dW} = \eta e_k^T \frac{dy_k}{dW},$$

where

$$\frac{dy_k}{dW} = \frac{\partial y_k}{\partial W} + \sum_{i=0}^{n} \frac{\partial y_k}{\partial x_{k-i}} \frac{dx_{k-i}}{dW} + \sum_{i=1}^{m} \frac{\partial y_k}{\partial y_{k-i}} \frac{dy_{k-i}}{dW}.$$
Dynamic Decoupled EKF

- Model weights as state of constant system driven by noise
  \[ W_{k+1} = W_k + v_k, \]
  \[ d_k = f(x_k, x_{k-1}, \ldots, y_{k-1}, y_{k-2}, \ldots W) + e_k. \]

- Extended Kalman Filter learning applied to learn “state” of system \(\Rightarrow\) The weights of the neural network.

- Update equations in full paper.
  - Use RTRL to calculate \( H(k) = dy_k^T/dW. \)
Adaptive Inverse Control

- Adapt $\hat{P}(z)$ to model $P(z)$. \textit{(Adaptive system ID)}
- Adapt $C(z)$ to control $\hat{P}(z)$. \textit{(Adaptive feedforward control)}
- Adapt $X(z)$ to cancel disturbances. \textit{(Adaptive disturbance canceling)}
Adaptive System Identification

- A plant model $\hat{P}(z)$ is adapted to minimize the mean-square difference between the plant and model outputs for the same input.

- The model is unbiased by disturbance if $u_k$ is statistically independent of $w_k$.

![Diagram showing the system where the plant and model are compared and adapted to minimize the mean-square difference.](Diagram)
Adaptive Feedforward Control

\[ u_k = g(u_{k-1}, u_{k-2} \ldots u_{k-m}, r_k, r_{k-1} \ldots r_{k-q}, W_C) \]
\[ y_k \approx \hat{y}_k = f(\hat{y}_{k-1}, \hat{y}_{k-2} \ldots \hat{y}_{k-n}, u_k, u_{k-1} \ldots u_{k-p}) \]

- Use the “backprop through (plant) model” (BPTM) algorithm to update \( W_C \).
Derivative Update Equations

\[
\frac{d u_k}{d W_C} = \frac{\partial u_k}{\partial W_C} + \sum_{j=1}^{m} \left( \frac{\partial u_k}{\partial u_{k-j}} \right) \left( \frac{d u_{k-j}}{d W_C} \right)
\]

\[
\frac{d \hat{y}_k}{d W_C} = \sum_{j=0}^{p} \left( \frac{\partial \hat{y}_k}{\partial u_{k-j}} \right) \left( \frac{d u_{k-j}}{d W_C} \right) + \sum_{j=1}^{n} \left( \frac{\partial \hat{y}_k}{\partial \hat{y}_{k-j}} \right) \left( \frac{d \hat{y}_{k-j}}{d W_C} \right).
\]

- Compact matrix implementation in full paper.
- Once computed, \( d \hat{y}_k / d W_C \) may be used with either BPTM-RTRL, or with BPTM-DDEKF.
- Disturbance cancellation similar.
Integrated MIMO System

- Full system shown to right.
- Three adaptive filters, one filter copy.
- Adaptation may occur concurrently.
**Illustrative Example**

- Plant dynamics (where $w_k$ is AWGN filtered by one-pole filter with pole at $z = 0.99$):
  \[
  s_k = \frac{s_{k-1}}{1 + s_{k-1}^2} + \sin(u_{k-1})
  \]
  \[
  y_k = s_k + w_k.
  \]

- $\mathcal{N}_{(3,1):5,1}$ neural networks trained to model plant.
- $\mathcal{N}_{(5,3):10,1}$ neural networks trained to perform feedforward control.
- $\mathcal{N}_{([5,3],1):10,1}$ neural networks trained to cancel disturbance.
- Comparison of algorithms based on convergence speed.
Feedforward Control Results

![Graphs showing the comparison of squared error over iterations for BPTM–RTIL feedforward–controller adaptation and BPTM–DDEKF feedforward–controller adaptation.]
Disturbance Cancellation Results

BPTM–RTLR disturbance–canceler adaptation

BPTM–DDEKF disturbance–canceler adaptation

## Tabulated Results

- Approximate number of iterations before convergence reached:

<table>
<thead>
<tr>
<th>System Type</th>
<th>(BPTM-) RTRL</th>
<th>(BPTM-) DDEKF</th>
</tr>
</thead>
<tbody>
<tr>
<td>System Identification</td>
<td>$1 \times 10^7$</td>
<td>$1 \times 10^5$</td>
</tr>
<tr>
<td>Feedforward Control</td>
<td>$1 \times 10^7$</td>
<td>$1 \times 10^5$</td>
</tr>
<tr>
<td>Dist. Cancellation</td>
<td>$1 \times 10^7$</td>
<td>$1 \times 10^4$</td>
</tr>
</tbody>
</table>

- DDEKF consistently one (or more) orders of magnitude faster than RTRL.
Conclusions

- Nonlinear controllers may be easily designed using AIC.
- Training these controllers with RTRL and BPTM-RTRL is very slow.
- DDEKF and BPTM-DDEKF have been implemented, and show consistent improvement in learning speed.
- Complexity of DDEKF is not much higher than RTRL.
- DDEKF appears to be the better choice.