Inference and control of electric loads given sparse measurements and communications delays

Johanna L. Mathieu
Department of Electrical Engineering & Computer Science
University of Michigan

Greg Ledva (UM), Laura Balzano (UM), Evangelos Vrettos (ETH), Silvia Mastellone (ABB), Göran Andersson (ETH)
Benefits and Challenges of the Modern Electric Grid

- Grid sensing and communication systems are becoming more prevalent
  - Cost & privacy concerns
  - Need methods to infer grid/load information from existing measurements

- Renewable energy resources are also becoming more prevalent
  - Most (e.g., wind and solar) are intermittent and uncertain
  - Need new sources of power system reserves
Overview

• **Inference:** Inferring the behavior of distributed energy resources with sparse measurements
  [Ledva, Balzano, & Mathieu *Allerton* 2015]

• **Control:** Controlling distributed electric loads to provide power system services with sparse measurements and input/measurement delays
  [Ledva, Vrettos, Mastellone, Andersson & Mathieu *HICSS* 2015]
  [Ledva & Mathieu *CDC (in review)* 2016]
Why do we want to disaggregate resources at the substation?

- Energy efficiency via conservation voltage reduction
- Contingency planning
- Optimal reserve contracting
- Demand response event signaling
- Demand response bidding
- Load coordination feedback
Disaggregation methods

- State estimation
  - Linear techniques require LTI system models
  - Nonlinear techniques can be computationally demanding

- Online learning
  - Optimization formulations
  - Model-free

- Hybrid approach: Dynamic Mirror Descent [Hall & Willet 2015]
  - Admits dynamic models of arbitrary forms
  - Optimization-based method to choose a weighted combination of the estimates of a collection of models
Outine: Part 1

• Dynamic Mirror Descent
• Problem setting: Plant data/models
• Algorithm Models
• Results
• Next steps
Dynamic Mirror Descent

- Mirror Descent: online algorithm to estimate a fixed state
- Dynamic Mirror Descent: online algorithm to estimate a dynamic state using a *collection of models* [Hall & Willet 2015]
  1. Compute the error between the model predictions and the measured data (i.e., loss function)
  2. Update the state in the direction of the negative gradient of the loss function

\[
\tilde{\theta}_t^i = \arg \min_{\theta \in \Theta} \eta_t \left\langle \nabla \ell_t(\tilde{\theta}_t^i, y_t), \theta \right\rangle + D \left( \theta \parallel \tilde{\theta}_t^i \right)
\]
Dynamic Mirror Descent

3. Use the estimated states to evaluate the models for the next time step
   \[ \hat{\theta}_{t+1}^i = \Phi_t^i(\tilde{\theta}_t^i) \]

4. Compute a weighted version of the estimates
   \[ \hat{\theta}_{t+1} = \sum_{i=1}^{N_{mdl}} w_{t+1}^i \hat{\theta}_{t+1}^i. \]

5. Update the model weights
   \[ w_{t+1}^i = \frac{\lambda}{N_{mdl}} + (1 - \lambda) \cdot \frac{w_t^i \exp \left( -\eta^r \ell_t \left( \hat{\theta}_t^i, y_t \right) \right)}{\sum_{j=1}^{N_{mdl}} w_t^j \exp \left( -\eta^r \ell_t \left( \hat{\theta}_t^j, y_t \right) \right)} \]
Problem Setting: Plant Data/Models

- Uncontrollable loads: data from Pecan Street Inc. Dataport
- Controllable loads: three-state hybrid models of air conditioners [Sonderegger 1978]
Algorithm Models: Uncontrollable loads

![Graph showing time of day against amplitude in MW, with data for different days of the week: y_{t}^{uc}, y_{t}^{uc},Mon, y_{t}^{uc},Tues, y_{t}^{uc},Wed.](image)
Algorithm Models: Controllable loads

- **Two-state hybrid models of air conditioners** [Mortensen & Haggerty 1988]
  - Temperature and ON/OFF mode

- **Sets of Linear Time Invariant (LTI) aggregate system models** [Mathieu et al. 2013]
  \[
  x^i_{t+1} = A^i x^i_t \\
  \tilde{y}^i_{t} = C^i x^i_t
  \]
  \[
  i \in \mathbb{N}^{\text{temps}}, \quad i \in \mathbb{N}^{\text{temps}}.
  \]

- **Sets of Linear Time Varying (LTV) aggregate system models**
  \[
  x_{t+1} = A_t x_t \\
  \tilde{y}^c_{t} = C_t x_t.
  \]
Algorithm Models: Controllable loads

- Two-state hybrid AC models do not work well.

![Graph showing Amplitude vs Time of Day with two lines representing different models.](image-url)
Algorithm Models: Controllable loads

- LTV models work better.
Results

Model bank is all combinations of uncontrollable and controllable load models (57 models)
Results: Weightings

Time of Day

Weight [-]

12AM 6AM 12PM 6PM 12AM

Other Models

\(\hat{y}_t^{c,PA}\)

\(\hat{y}_t^{c,LTV1}\)

\(\hat{y}_t^{c,LTV2}\)

\(\hat{y}_t^{c,LTI,36}\)
Results: Bad Models

- All uncontrollable load models are too low.
### Results: Summary

<table>
<thead>
<tr>
<th>Case</th>
<th>RMS Error (kW)</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Benchmark:</strong> Use current outdoor temperature to evaluate simple controllable load model</td>
<td>738</td>
</tr>
<tr>
<td><strong>DMD Case 1:</strong> Includes every combination of uncontrollable and controllable models</td>
<td>264</td>
</tr>
<tr>
<td><strong>DMD Case 2:</strong> Case 1 models plus a smoothed version of the actual uncontrollable load</td>
<td>211</td>
</tr>
<tr>
<td><strong>DMD Case 3:</strong> Case 2 models plus more accurate models of the controllable load over time periods where the other models are less accurate</td>
<td>175</td>
</tr>
<tr>
<td><strong>DMD Case 4:</strong> Includes uncontrollable load models that underestimate the uncontrollable load</td>
<td>1392</td>
</tr>
</tbody>
</table>
Recall: $w_t^{i+1} = \frac{\lambda}{N_{mdl}} + (1 - \lambda) \cdot \frac{w_t^{i} \exp(-\eta^r \ell_t(\hat{\theta}_t^{i}, y_t))}{\sum_{j=1}^{N_{mdl}} w_t^{j} \exp(-\eta^r \ell_t(\hat{\theta}_t^{j}, y_t))}$

Results: Varying Algorithm Parameters

![Graph showing RMS Error vs. \( \eta^r \)]

- Under-fitting
- Over-fitting
Next steps

• Investigate more realistic settings (using more real data)
• Develop better load models
• Improve the algorithm, e.g., alternative weighting functions
• Investigate identifiability
• Incorporate additional measurements (reactive power, voltage) into the approach
Key findings

• Dynamic Mirror Descent (DMD) enables us to solve the substation disaggregation problem leveraging dynamical models of arbitrary form

• DMD can work well (on simple examples); however, it is easy to find instances where it does not work well
Overview

• **Inference:** Inferring the behavior of distributed energy resources with sparse measurements
  [Ledva, Balzano, & Mathieu *Allerton* 2015]

• **Control:** Controlling distributed electric loads to provide power system reserves with sparse measurements and input/measurement delays
  [Ledva, Vrettos, Mastellone, Andersson & Mathieu *HICSS* 2015]
  [Ledva & Mathieu *CDC (in review)* 2016]
How can loads provide reserves?

→ your refrigerator is already flexible

![Diagram showing normal operation and non-disruptive load control](image)

normal operation

non-disruptive load control
Thousands of thermostatically controlled loads (TCLs) can track signals and provide reserves.

TCLs: air conditioners, heat pumps, space heaters, electric water heaters, refrigerators.

[Mathieu, Koch, and Callaway IEEE Transactions on Power Systems 2013]
Simulation results:
1000 ACs tracking 5-minute market signal

Controller gets temperature/state of each load every 2 seconds

Controller infers TCL behavior from power measurements at the substation

The more the controller knows about the loads, the better it can track a signal

[Mathieu, Koch, and Callaway IEEE Transactions on Power Systems 2013]
Data from loads

• Parameters
  – the make/model of the load?
  – its temperature setpoint/dead-band width?
  – some information about the household?

• Real-time data
  – Measurements of the on/off state and/or internal temperature?
  – Household smart meter data?
  – Power measurements from the distribution network?

• Recorded data
  – high resolution power measurements of each load?
Communication and control scenario

Load aggregator

Infrequent state measurements

Frequent aggregate power (output) measurements

[Ledva, Vrettos, Mastellone, Andersson, & Mathieu HICSS 2015]
Delays cause unsynchronized arrivals of inputs at the loads and measurements at the controller.
The challenge

• Design an estimator and controller to enable loads to track a signal *despite delays*

• Assuming...
  – Control inputs & measurements are time-stamped
  – Delay statistics are known
  – State measurements are taken frequently; measurement *histories* are transmitted infrequently
  – Aggregate power measurements are *very* noisy (though the noise is normally distributed, zero-mean, and the standard deviation is known)
Two-state TCL model

Each TCL $i$ is modeled with a stochastic hybrid difference equation:

**Temperature of the space**

$$\theta_i(k + 1) = a_i \theta_i(k) + (1 - a_i) (\theta_{a,i} - m_i(k) \theta_{g,i}) + \epsilon_i(k)$$

**On/off state**

$$m_i(k + 1) = \begin{cases} 
0, & \theta_i(k + 1) < \theta_{set,i} - \delta_i/2 \\
1, & \theta_i(k + 1) > \theta_{set,i} + \delta_i/2 \\
m_i(k), & \text{otherwise}
\end{cases}$$

- $a$, thermal parameter
- $\theta_g$, temperature gain
- $\theta_a$, ambient temperature
- $\epsilon$, noise
- $\theta_{set}$, set point
- $\delta$, dead-band width

\[ x(k + 1) = Ax(k) + Bu(k) + B_\omega \omega(k) \]
\[ y(k) = Cx(k) + \nu(k) \]

Similar models in the literature:
- Lu & Chassin 2004/2005
- Bashash & Fathy 2011/2013
- Kundu & Hiskens 2011
- Zhang et al. 2013

[Mathieu, Koch, and Callaway *IEEE Transactions on Power Systems* 2013]
Estimator Designs

• Based on Kalman Filtering
  – Estimator 1: Parallel filter estimator
    • One Kalman Filter per load
    • Each time a measurement arrives, filter it
    • Synthesize aggregate estimate from individual estimates
  – Estimator 2: Single Kalman Filter Using Aggregate State Predictions
    • Use state measurement histories to estimate *individual* load parameters (two-state model)
    • Use individual load models to predict current state
    • Use predictions as “measurements” in Kalman Filter
Based on Model Predictive Control

- Use knowledge of delay distributions and past control inputs

First control sequence: \(u_1, u_2, u_3, \ldots, u_n\)

Second control sequence: \(u_2, u_3, \ldots, u_{n+1}\)

Third control sequence: \(u_3, u_4, \ldots, u_{n+1}\)

Input estimate: \(\hat{u}_k = U_k P\)
Control Formulation

\[
\begin{align*}
\text{minimize} \quad & \sum_{k=t}^{t+N_{\text{mpc}}-1} \left[ c^y (y_k^\text{err})^2 + c^\delta (\delta_k^- + \delta_k^+) + c^u (u_k|m) \right], \\
\text{s.t.} \quad & x_{k+1} = A x_k + B \hat{u}_k, \\
& \hat{u}_k = \mathcal{U}_k \mathcal{P}, \\
& y_k^\text{err} = y_k^\text{P,ref} - C^P x_k, \\
& u_k|m \leq x_k^i, \\
& -u_k|m \leq x_k^{N_x+1-i}, \\
& 0 - \delta_k^- \leq x_k \leq 1 + \delta_k^+, \\
& 0 \leq \delta_k^-, \delta_k^+. 
\end{align*}
\]

Tracking error \quad State constraint deviations 
Control effort
Case Studies

- PJM Regulation Signals, Reg-A & Reg-D
- Average input delay of 20 seconds
- (Delayed) state histories arrive every 15 minutes
Results

![Graph showing demand change over time for different estimators and reference signals.]

- Estimator 1
- Estimator 2
- Reference Signal

Demand Change (%)

Time (hr)

Reg-A

Reg-D
Results: Model Mismatch

![Graph showing demand change over time for different estimators and reference signals.](image)
Control Results: Reg-A

Two-State Plant

Three-State Plant

RMSE (%)

0 0.25 0.75 1.0 1.5 2.0 2.5 3.0 3.5 4.0

PJM Score (-)

0 0.25 0.75 1.0 1.5 2.0 2.5 3.0

Average Delay (sec.)

0 10 20

Estimator 1

Estimator 2
Control Results: Reg-D

Two-State Plant

Three-State Plant

RMSE (%)

PJM Score (-)

0 10 20
Average Delay (sec.)

0 0.25 0.75 1

Estimator 1
Estimator 2

4/28/16
J. Mathieu, UMich
Controller Reformulation

Original Model

\[ x_{k+1} = A x_k + B u_k \]
\[ y_k = C x_k. \]

Modal Model

\[
\begin{bmatrix}
1 \\
\tilde{x}_{k+1}
\end{bmatrix} =
\begin{bmatrix}
1 & 0 \\
0 & \tilde{A}
\end{bmatrix}
\begin{bmatrix}
1 \\
\tilde{x}_k
\end{bmatrix} +
\begin{bmatrix}
0 \\
\tilde{B}
\end{bmatrix} u_k
\]
\[ y_k = \begin{bmatrix} y_{ss} & \tilde{C} \end{bmatrix} \begin{bmatrix} 1 \\
\tilde{x}_k
\end{bmatrix} \]

Reduced-Order Model

\[ \tilde{x}_{k+1} = \tilde{A} \tilde{x}_k + \tilde{B} u_k \]
\[ \tilde{y}_k = \tilde{C} \tilde{x}_k. \]
Controller Reformulation

The linear controller uses constant gains generated from an output-regulating Linear Quadratic Regulator (LQR) with reference feedforward.

**Linear Controller**

\[ u_{t}^{\text{seq}} = -K_{\infty}^{x} \bar{x}_{t} - K_{\infty}^{w} w_{t} + K_{\infty}^{y} y_{t}^{\text{des}} \]

**LQR Formulation**

\[
\begin{align*}
\text{min} & \quad \sum_{k=t}^{\infty} \begin{bmatrix} \bar{x}_{k} \\ w_{k} \end{bmatrix}^{T} \begin{bmatrix} \bar{C}^{T} & q^{y} \bar{C} & 0 \\ 0 & q^{w} \end{bmatrix} \begin{bmatrix} \bar{x}_{k} \\ w_{k} \end{bmatrix} + (u_{k}^{\text{seq}})^{T} R u_{k}^{\text{seq}} \\
\text{s.t.} & \quad \begin{bmatrix} \bar{x}_{k+1} \\ w_{k+1} \end{bmatrix} = \begin{bmatrix} \bar{A} & 0 \\ \bar{C} & 0 \end{bmatrix} \begin{bmatrix} \bar{x}_{k} \\ w_{k} \end{bmatrix} + \begin{bmatrix} \bar{B} \\ 0 \end{bmatrix} u_{k}^{\text{seq}} \\
\end{align*}
\]

**Feedforward Gain**

\[ K_{\infty}^{y} = \left( \bar{C} \{ zI - \bar{A} + \bar{B} K_{\infty}^{x} \}^{-1} \bar{B} \right)^{-\dagger} \]
Case Studies

- PJM Regulation Signals, Reg-A & Reg-D
- Average input delay of 20 seconds
- Full state feedback, no measurement delay
- Three-state models used for the plant
In this paper, we developed a linear feedback controller that mitigates the effect of input delays, and we compared it to a linear feedback controller in conjunction with an estimator that addresses reduced computational complexity and provides a closed-form controller. Also, the linear feedback controller benefits from knowledge about the input delay statistics. The linear feedback controller improves each time-step and by incorporating knowledge about the input delays by generating an open-loop input sequence at real frequency regulation signals. Both methods counteract the effect of input delays, and we compared it to a linear feedback controller.
The number of state bins and amplitude of the reference signal also influence the results; situations where the input may be achievable by additional tuning. The MPC and LIN provides sample time series. Note that the controllers were explored this scenario, we increase the amplitude of the reference signal to 80% of the mean steady-state TCL demand. Alternatively, MPC achieves 0.83% and no constraint violations.

When tracking the “Reg-A” Reference without delays, LIN took 0.0041 seconds on average and a maximum of 0.0079 seconds. The computation reduction in LIN took 0.1095 seconds on average and a maximum of 0.1204 seconds. The number of state bins and amplitude of the reference signal also influence the results; situations where the input may be achievable by additional tuning. The MPC and LIN provides sample time series. Note that the controllers were explored this scenario, we increase the amplitude of the reference signal to 80% of the mean steady-state TCL demand. Alternatively, MPC achieves 0.83% and no constraint violations.

When tracking the “Reg-A” Reference without delays, LIN took 0.0041 seconds on average and a maximum of 0.0079 seconds. The computation reduction in LIN took 0.1095 seconds on average and a maximum of 0.1204 seconds. The number of state bins and amplitude of the reference signal also influence the results; situations where the input may be achievable by additional tuning. The MPC and LIN provides sample time series. Note that the controllers were explored this scenario, we increase the amplitude of the reference signal to 80% of the mean steady-state TCL demand. Alternatively, MPC achieves 0.83% and no constraint violations.

When tracking the “Reg-A” Reference without delays, LIN took 0.0041 seconds on average and a maximum of 0.0079 seconds. The computation reduction in LIN took 0.1095 seconds on average and a maximum of 0.1204 seconds. The number of state bins and amplitude of the reference signal also influence the results; situations where the input may be achievable by additional tuning. The MPC and LIN provides sample time series. Note that the controllers were explored this scenario, we increase the amplitude of the reference signal to 80% of the mean steady-state TCL demand. Alternatively, MPC achieves 0.83% and no constraint violations.

When tracking the “Reg-A” Reference without delays, LIN took 0.0041 seconds on average and a maximum of 0.0079 seconds. The computation reduction in LIN took 0.1095 seconds on average and a maximum of 0.1204 seconds. The number of state bins and amplitude of the reference signal also influence the results; situations where the input may be achievable by additional tuning. The MPC and LIN provides sample time series. Note that the controllers were explored this scenario, we increase the amplitude of the reference signal to 80% of the mean steady-state TCL demand. Alternatively, MPC achieves 0.83% and no constraint violations.

When tracking the “Reg-A” Reference without delays, LIN took 0.0041 seconds on average and a maximum of 0.0079 seconds. The computation reduction in LIN took 0.1095 seconds on average and a maximum of 0.1204 seconds. The number of state bins and amplitude of the reference signal also influence the results; situations where the input may be achievable by additional tuning. The MPC and LIN provides sample time series. Note that the controllers were explored this scenario, we increase the amplitude of the reference signal to 80% of the mean steady-state TCL demand. Alternatively, MPC achieves 0.83% and no constraint violations.

When tracking the “Reg-A” Reference without delays, LIN took 0.0041 seconds on average and a maximum of 0.0079 seconds. The computation reduction in LIN took 0.1095 seconds on average and a maximum of 0.1204 seconds. The number of state bins and amplitude of the reference signal also influence the results; situations where the input may be achievable by additional tuning. The MPC and LIN provides sample time series. Note that the controllers were explored this scenario, we increase the amplitude of the reference signal to 80% of the mean steady-state TCL demand. Alternatively, MPC achieves 0.83% and no constraint violations.

When tracking the “Reg-A” Reference without delays, LIN took 0.0041 seconds on average and a maximum of 0.0079 seconds. The computation reduction in LIN took 0.1095 seconds on average and a maximum of 0.1204 seconds. The number of state bins and amplitude of the reference signal also influence the results; situations where the input may be achievable by additional tuning. The MPC and LIN provides sample time series. Note that the controllers were explored this scenario, we increase the amplitude of the reference signal to 80% of the mean steady-state TCL demand. Alternatively, MPC achieves 0.83% and no constraint violations.

When tracking the “Reg-A” Reference without delays, LIN took 0.0041 seconds on average and a maximum of 0.0079 seconds. The computation reduction in LIN took 0.1095 seconds on average and a maximum of 0.1204 seconds. The number of state bins and amplitude of the reference signal also influence the results; situations where the input may be achievable by additional tuning. The MPC and LIN provides sample time series. Note that the controllers were explored this scenario, we increase the amplitude of the reference signal to 80% of the mean steady-state TCL demand. Alternatively, MPC achieves 0.83% and no constraint violations.
Key takeaways

• Communication network limitations necessitate controller/estimator designs that cope with delays, bandwidth limitations, etc.

• Delays make loads less capable of providing fast services, but we can mitigate these impacts through delay-aware control and estimation.
Conclusions

• Need methods to infer electric load behavior from existing measurements
  – Dynamic mirror descent applied to distribution substation measurements

• Need new sources of power system reserves
  – Coordination of distributed electric loads using delay-aware control/estimation

Funding: NSF Grant ECCS-1508943.