NREL
NATIONAL RENEWABLE ENERGY LABORATORY

Distributed Reinforcement Learning with ADMM-RL

Peter Graf, Jen Annoni, Chris Bay, Devon Sigler, Dave Biagioni, Monte Lunacek, Andrey Bernstein, Wesley Jones

Innovative Optimization and Control Methods for Highly Distributed Autonomous Systems
April 11-12, 2019
Golden, Colorado

Distributed Reinforcement Learning with ADMM-RL

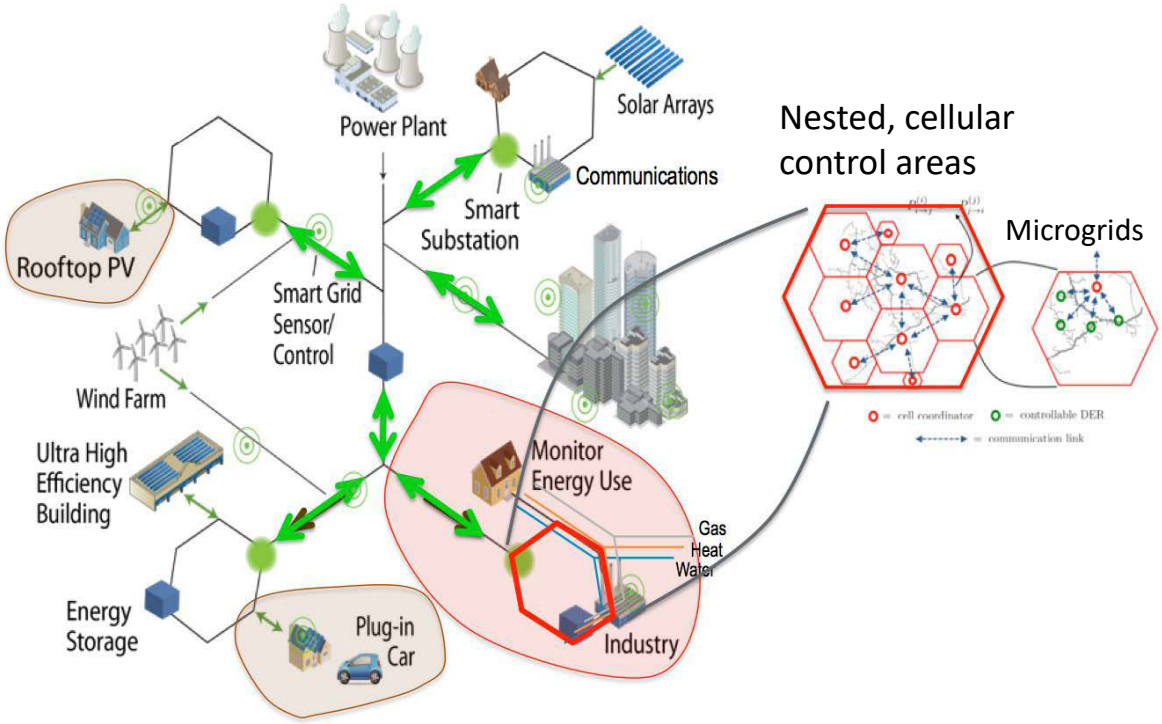
- Context: Autonomous Energy Systems
- Reinforcement Learning (RL): an AI control strategy
  - Control of nonlinear systems over multi-step time horizons learned by experience,
  - Not computed online by optimization.
- Alternating Direction Method of Multipliers (ADMM): a distributed control meta-algorithm
  - dual decomposition (enables decoupled, parallel, distributed solution)
  - method of multipliers (enables convexification/stability/convergence).
  - Iterates among subproblems.
- ADMM-RL combines ADMM and RL
  - Integrates learned controllers as subsystems in distributed control.
  - replaces subproblems in ADMM with RL.
- Benefits
  - (RL alone) Enables “over-time” control of nonlinear systems
  - Resulting pretrained sub-solver speeds up deployed distributed controller.
- We illustration ADMM-RL in
  - distributed wind farm yaw control, and
  - distributed grid-aware demand aggregation for water heaters.

Main points

- Energy systems rapidly becoming too complex to control optimally via real-time optimization.
- Reinforcement learning has potential to bypass online optimization and enable control of highly nonlinear stochastic systems.
- ADMM-RL extends RL to distributed control context.
- RL as an additional strategy within distributed control is a very interesting concept (e.g., top-down vs bottom-up; what is autonomy?)

Background: Autonomous Energy Systems

Beyond central-station based grid



Challenges of the Future Grid

- Going from hundreds to millions of controllable assets (both large-scale central station control and individual microgrid control has been accomplished – but we have not accomplished fully linking from large to small scales)
- Increasing data and information from sensors
- Unable to use current optimization techniques because of computational intractability
- More interdependencies with communications and other energy domains (heat/cooling, gas, water, transportation)

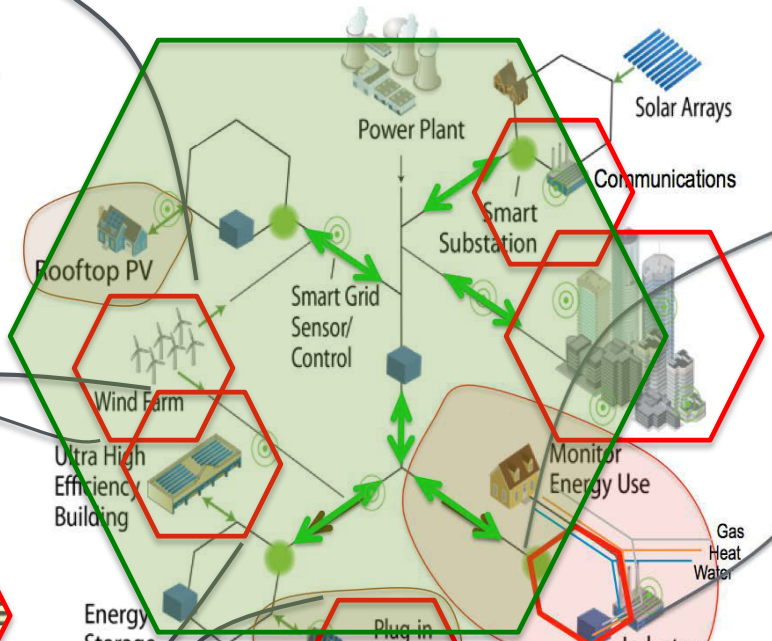
Key Features of AES

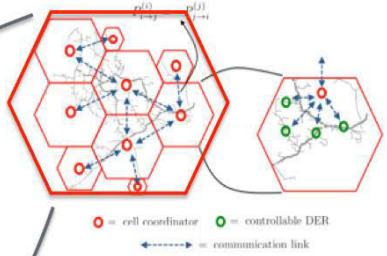
- **Autonomous** – Makes decisions without operators
- **Resilient** – Self-reconfiguring, cellular building blocks, able to operate with and without communications
- **Secure** – Incorporates cyber and physical security against threats
- **Reliable and Affordable** - Self optimizes for both economics and reliability
- **Flexible** – Able to accommodate energy in all forms including variable renewables

Courtesy Ben Kroposki

Autonomous Energy Systems

Wind Plant Controls to AES





Buildings to AES

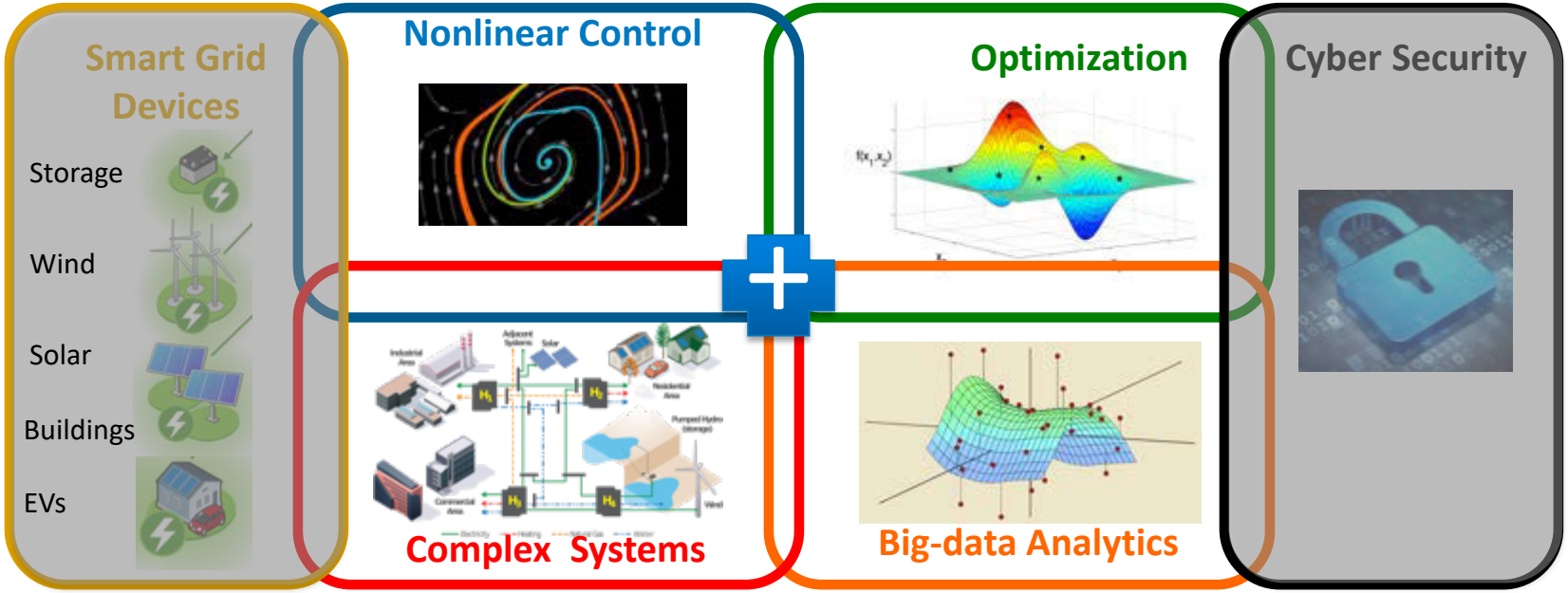
Vehicle to AES

Research Needs in Autonomous Energy Systems

- Equivalent in idea to autonomous vehicles, *"Autonomous Energy Grids"* do not require operators and make independent decisions. They can self-reconfigure and optimize themselves for reliability and economic performance while integrating energy in all forms

- Need to advance foundational science:
  - Smart Device/ Power Electronics
  - Cybersecurity
  - Non-linear Control Theory
  - Optimization Theory
  - Complex System Theory
  - Big Data Analytics

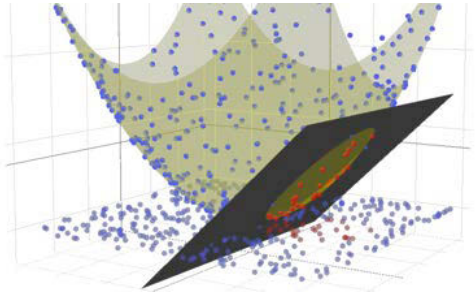
Need to develop new mathematical formulations and a common analytical framework for modeling, optimization, and control of complex systems at multiple spatial and temporal scales



AES Priority Research Directions

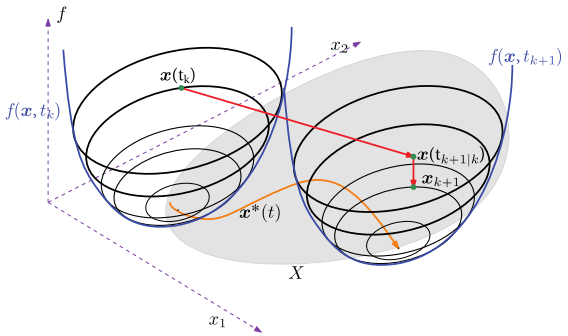
Big Data Analytics

- Develop ways to use heterogeneous grid data (addressing access and privacy) to better conduct ensemble forecasting of grid states and enable automated and distributed decision making from machine learning techniques.



Optimization Theory

- Develop computationally-affordable, stable, and provably optimal algorithms that can be implemented in real-time and distributed fashions.

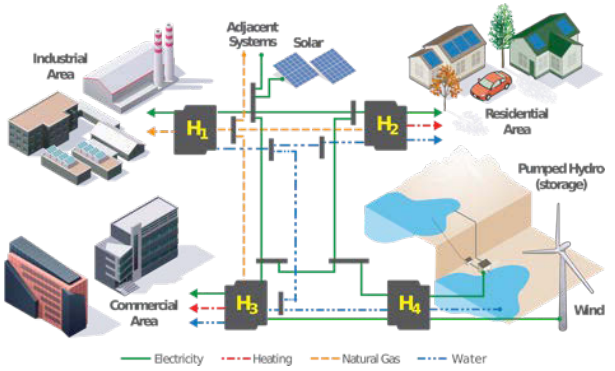


Controls Theory

- Develop scalable, real-time, decentralized and distributed controls that take into account inherently asynchronous operations as a result of communications delays, losses, and distributed (asynchronous) control actions.

Complex Systems Theory

- Develop modeling and simulation methods that address integration and interdependencies of many different energy and communications systems at various temporal and spatial scales.



Motivation/Perspective for AI in AES

- **Coordination of devices is needed in order to enable high penetration of renewables economically.**
- **Enabling grid operation is not the primary function of many systems**
  - Building equipment (e.g. water heaters), electric vehicles, wind turbines, solar inverters, batteries,… all have their own goals.
- **The systems are complicated**
  - Nonlinearity, model uncertainty, stochasticity, nonconvexity, time-dependence
  - Whole system control requires solution of large online optimization of approximated problems.
- **AI aims for controllers that**
  - *learn* from *experience* to operate in complex environments,
  - adapt to stochastic input and uncertain feedback,
  - avoid online optimization,
  - *react* rather than *think.*
- **Our current emphasis is on scalable integration of AI and traditional methods.**

Test domain 1: wind farm yaw control

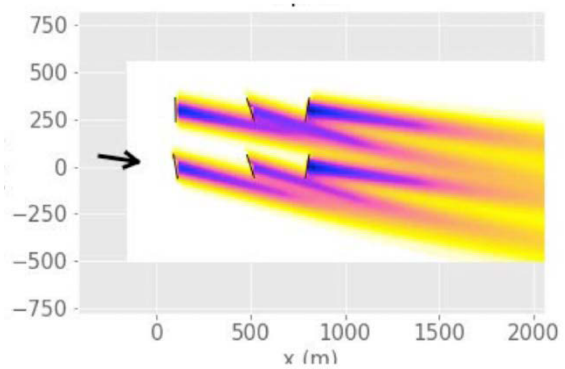
Downwind turbines generate less energy

Goal: yaw the turbines to optimally steer the wakes

$$P_{tot}(x) = \sum_{i}^{Nturb} P_i(x)$$

State of the art is NREL tool FLORIS:

- Nonlinear physics simulation
- Steady state optimization



Our model problem is over time:

$$P_{tot}(x) = \sum_{j}^{Ntimes} \sum_{i}^{Nturb} P_i(x^j)$$

$$\min_{x} P_{tot}(x)$$

Test domain 2: grid aware water heater control

https://smartenit.com/portfolio-items/iot_water_heater/

Water heaters can be used as energy storage, especially if coordinated on large scale

Demand response devices:

selfish (autonomous) objectives

$f(x_i) =$ Cold shower penalty + energy cost

cooperative (connected) objectives

$g(\sum_i x_i) =$ Power overuse penalty

x_i = set point of water heater i.

Linear physics

"Competing" solvers:

- Mixed Integer Non Linear Program
- Model Predictive Control

As above, goal is optimal behavior **over time** horizon:

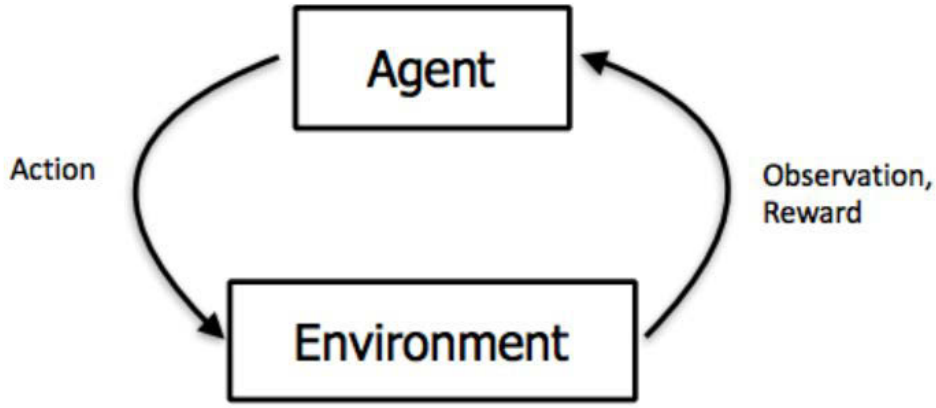
$$\min_x \sum_{j=1}^{N_{times}} \sum_i f(x_i^j) + g(\sum_i x_i^j)$$

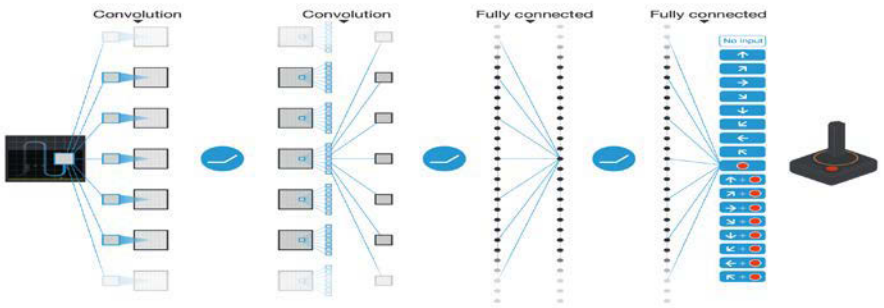
AI for sequential decisions: Reinforcement Learning (RL)

RL allows for evolving correct actions based on experience: learning by doing.

Applications (press & papers):

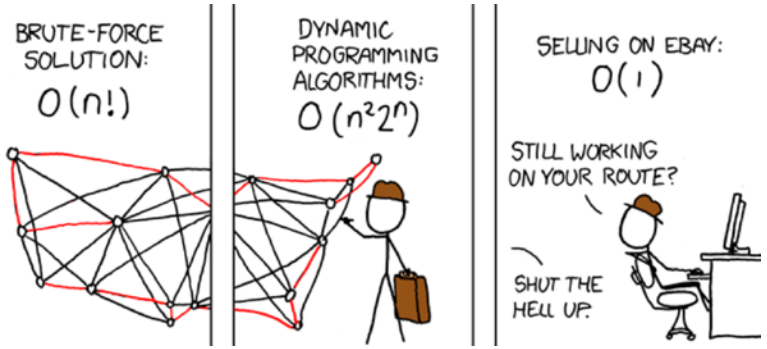
- Games--Alpha Go, Atari
- Robotics
- Web System Configuration
- Chemistry
- Personalized Recommendatior
- Bidding and Advertising
- Resources management in computer clusters
- Traffic Light Control
- Self Driving Cars
- Power Systems
- …





Introduction to Reinforcement Learning—traditional approach

Bellman's Principle of Optimality:
``An optimal policy has the property that whatever the initial state and initial decision are, the remaining decisions must constitute an optimal policy with regard to the state resulting from the first decision." (See Bellman, 1957, Chap. III.3.



States	s
Action	x
Reward [cost]	r(s) [c(s)]
Transition	s'=T(s,x), or P(s'|s,x)
Discount factor	γ
Policy	π
Value Function	V(s)
Q-Function	Q(s,a)

$$Q^*(s,x) = r(s) + \gamma \sum_{x'} P(s'|s,x') \min_{x'} Q^*(s',x')$$

$$Q^*(s,x) = r(s) + \gamma \min_{x'} Q^*(s',x')$$

$$x = \pi(s) \qquad s' = T(s,\pi(s))$$

TD learning:

$$Q^{k+1}(s,x) = (1-\alpha)Q^k(s,x) + \alpha\{r(s) + \gamma \min_{x'} Q^k(s',x')\}$$

"DQN": (sort of supervised) train neural network so that

$$Q_{\theta_i}(s_i,x_i) \Rightarrow r(s_i) + \gamma \min_{x_i'} Q_{\theta_{i-1}}(s_i',x_i')$$

RL with linear policy and "random search"

Here we will use a MUCH simpler approach to RL

$$x = \pi(s) = \theta^T s$$ **LINEAR policy. Action is a linear function of state**

$$L(\theta) = \sum_{j}^{N_{times}} c(s^j)$$ **No Bellman equation, just optimize "episode" cost**

RL is "just" global optimization: minimize $L(\theta)$

Augmented Random Search (ARS):
"**Simple random search provides a competitive approach to reinforcement learning**" (https://arxiv.org/pdf/1803.07055.pdf)
B. Recht et al, UC Berkeley

~Algorithm computes finite difference directional derivatives in random directions and follows the most promising. ~

Introduction to ADMM

Boyd et al, "Distributed Optimization and Statistical Learning via the Alternating Direction Method of Multipliers"

Alternating Direction Method of Multipliers (ADMM) solves problems of the form:

$$\begin{aligned} &\text{minimize } f(x) + g(z) \\ &\text{s.t. } Ax + Bz = c \end{aligned}$$

Of particular interest for us is when f(x) is additive: $f(x) = \sum_i f_i(x)$

Origins:

Dual Decomposition – parallelism for constrained additive objectives

Method of Multipliers – improved convergence via smooth (quadratic) penalty

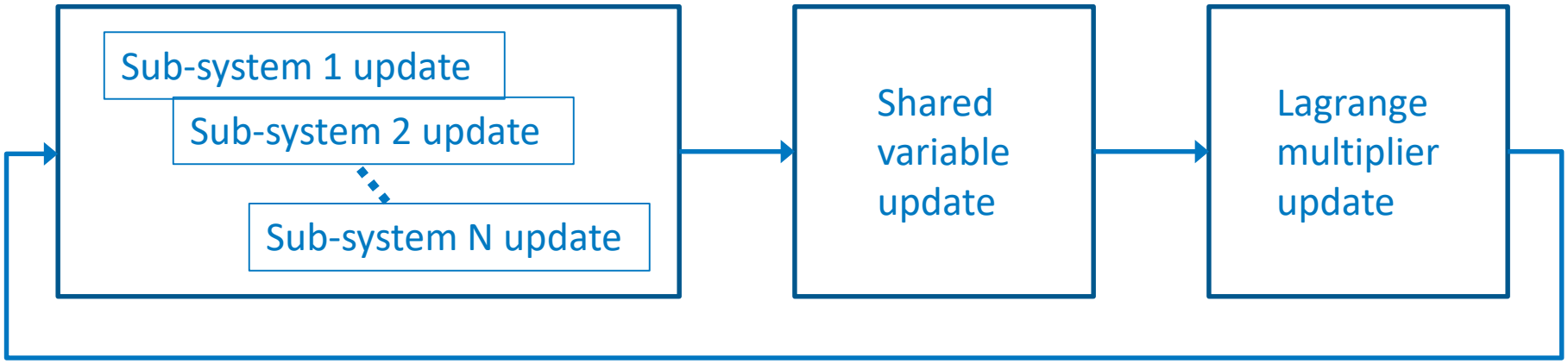
The method: Write "augmented Lagrangian"

$$L_p(x, z, y) = f(x) + g(z) + y^t(Ax + Bz - c) + (\rho/2)||Ax + Bx - c||_2^2$$

Solve iteratively:

$$\begin{aligned} x^{k+1} &= \underset{x}{\operatorname{argmin}}\, L_\rho(x, z^k, y^k) \\ z^{k+1} &= \underset{z}{\operatorname{argmin}}\, L_\rho(x^{k+1}, z, y^k) \\ y^{k+1} &= y^k + \rho(Ax^{k+1} + Bz^{k+1} - c) \end{aligned}$$

Structure of ADMM and ADMM-RL



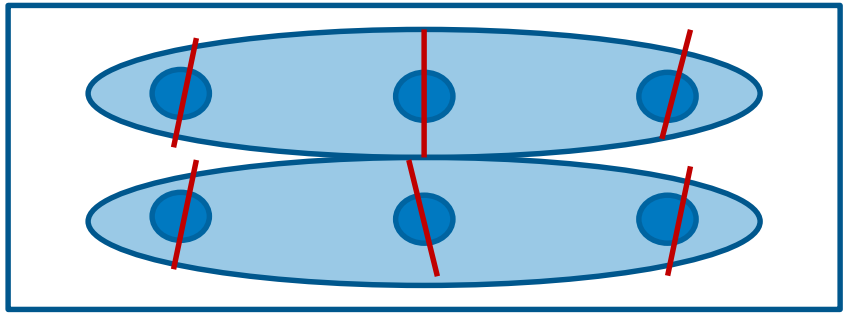
Common framework for both optimization and learning based distributed controllers:

- ADMM: Subsystem updates are explicit optimization.
- ADMM-RL training: Subsystems updates are some number of RL training iterations.
- ADMM-RL operating: Subsystem updates fill in optimal control action from learned RL controller.

ADMM “consensus”, e.g., distributed wind farm control

The consensus formulation: applicable to any objective function that can be written in the form

$$f(x) = \sum_{i}^{N} f_i(x)$$



To apply ADMM, we imagine N copies of x and introduce auxiliary variable z, at which point the problem can be written

$$\text{minimize} \sum_i f_i(x_i) + g(z) \text{ s.t. } x = z$$

Seemingly over-complex restatement of the problem, but:

- replace single large optimization problem with N smaller ones,
- and these can each be solved in parallel.

The price we pay is that now we have to iterate the process to self-consistency (``global consensus"); but this is frequently a worthwhile tradeoff.

We employ this “consensus” formulation for distributed wind farm yaw control.

ADMM for wind farm for yaw control

Wind farm yaw control problem is a consensus problem:

- Partitioning set of turbines into disjoint groups. *Intuition -- depending on wind direction, there is a natural partitioning into groups of turbines whose wakes affect each other strongly, with less strong wake interaction between groups*.

- There remains non-zero interaction *between* groups. The problem is not completely decoupled. Solve with ADMM by solving for each group independently, and iterating to self-consistency:

(Following Boyd, section 7.1, after employing several straightforward simplifications), wind turbine yaw control global consensus problem is

$$x_i^{k+1} = \underset{x_{p(i)}}{\operatorname{argmin}} f_i(x_i) + y_k^{k,T}(x_i - \bar{x}^k) + \frac{\rho}{2}||x_i - \bar{x}^k||^2$$

$$y^{k+1} = y^k + \rho(x_i^{k+1} - \bar{x}^{k+1})$$

Here, y is a Lagrange multiplier that enforces the global consensus, and ρ is the Lagrangian penalty parameter.
Overbars indicate averages over all the wind turbines
No “g” term.

ADMM “sharing”, e.g. grid-award demand response aggregation

The ``sharing" problem:

- x can be *partitioned* into sub-vectors x_i
- function f is a sum of terms f_i that only depend on x_i,
- overall objective contains an additional term g that is a function of ALL the component of x_i. That is, if we have

$$\text{minimize} \sum_i f_i(x_i) + g(\sum_i x_i)$$

where the x_i make up a partition of x,

we can ``implement" this in ADMM as

$$\text{minimize} \sum_i f_i(x_i) + g(z) \text{ s.t. } x = z$$

which allows for decoupling of the x_i minimization problems.

water heater example: ADMM updates of each iteration will involve

- minimization over x_i for each water heater separately and thus easily parallelizable—this stage optimizes each water heater for comfort and cost,
- a single minimization over z—this stage ensures we don’t use more power than is globally available.
- and a final update of the Lagrange multiplier that links x and z.

ADMM for water heater demand response aggregation

In scaled form, and having employed simplifications (as describe in BoydADMM, section 7.3), the ADMM algorithm is

$$x_i^{k+1} = \underset{x}{\operatorname{argmin}}\, f_i(x_i) + \frac{\rho}{2}||x_i - x_i^k + \bar{x}^k - \bar{z}^k + u^k||^2$$

$$\bar{z}_{k+1} = \underset{\bar{z}}{\operatorname{argmin}}\, g(N\bar{z}) + \frac{N\rho}{2}||\bar{z} - u^k - \bar{x}^{k+1}||^2$$

$$u^{k+1} = u^k + \bar{x}^{k+1} - \bar{z}^{k+1}$$

where u is the scaled Lagrange multiplier, ρ is the augmented Lagrangian parameter, and N is the number of water heaters.

Here we have used the fact that the collective power overuse function g, above, is actually a function of the average x_i, thus (assuming the condition x=z is met),

$$g(\sum_i^{N_{wh}} x_i) = g(\sum_i^{N_{wh}} z_i) = g(N\bar{z})$$

The symbol x^k_i is the *vector* whose components are the decisions at each time step.

ADMM-RL

- For **scalability** and **autonomy** of AES, distributed control is key.
- We have **developed a new mathematical technique** that combines two powerful methods: Alternating Direction Method of Multipliers (ADMM), and Reinforcement Learning (RL).
- It **has the potential benefit of** 1) enabling the construction of distributed controllers in the most complex cases; 2) increasing the efficacy of the deployed distributed devices by orders of magnitude.
- We have preliminary **demonstrations** of these benefits in 1) distributed wind farm yaw control, 2) demand response aggregation of a standard water heater model.

ADMM-RL Concepts

- **Running several steps of learning (e.g., ARS) to improve the policy is like *approximately* solving the ADMM subproblem.**
- **So let us *replace* one of the subproblems with some number of ARS iterations.**
- **An attractive combination:**
  - employ learning myopically in the targeted context of each subsystem on its own, and
  - employ formally convergent ADMM updates to achieve global convergence.

ADMM-RL steps

Consensus (e.g. wind farm)

$$x_i^{k+1} = \underset{x_{p(i)}}{\text{argmin-RL}} f_i(x_i) + y_k^{k,T}(x_i - \bar{x}^k) + \frac{\rho}{2}||x_i - \bar{x}^k||^2$$

$$y^{k+1} = y^k + \rho(x_i^{k+1} - \bar{x}^{k+1})$$

Sharing (e.g. water heater)

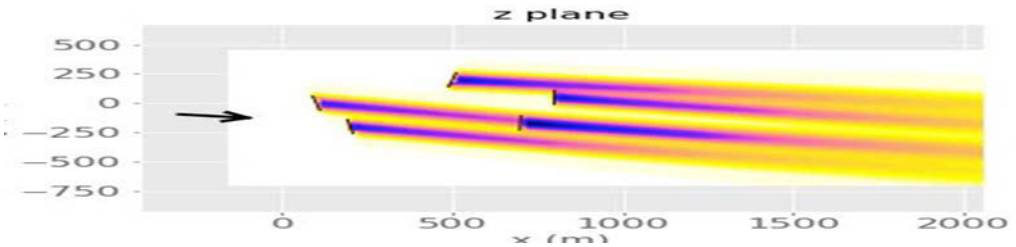
$$x_i^{k+1} = \underset{x}{\text{argmin-RL}} f_i(x_i) + \frac{\rho}{2}||x_i - x_i^k + \bar{x}^k - \bar{z}^k + u^k||^2$$

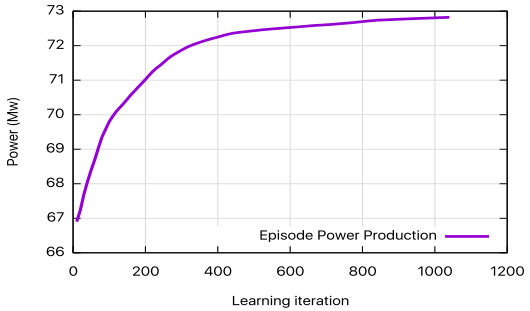
$$\bar{z}_{k+1} = \underset{\bar{z}}{\text{argmin}}\, g(N\bar{z}) + \frac{N\rho}{2}||\bar{z} - u^k - \bar{x}^{k+1}||^2$$

$$u^{k+1} = u^k + \bar{x}^{k+1} - \bar{z}^{k+1}$$

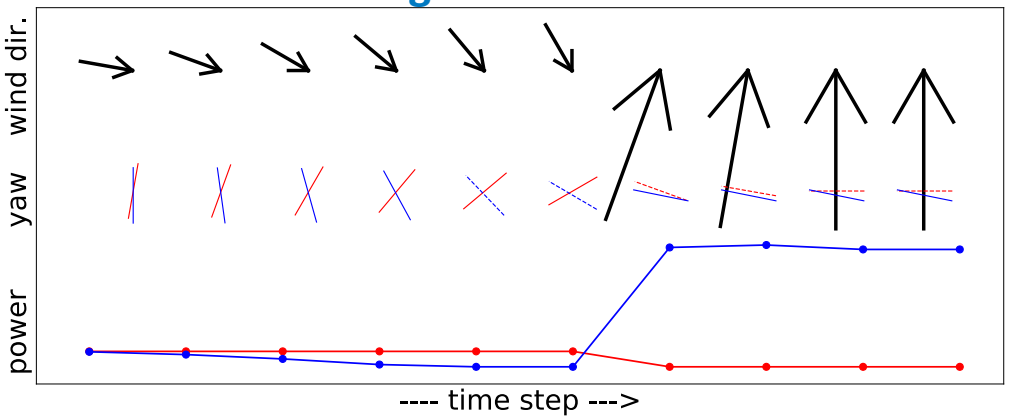
Wind farm 1: central RL and “tricking” FLORIS

A random 5 turbine configuration and its learning curve



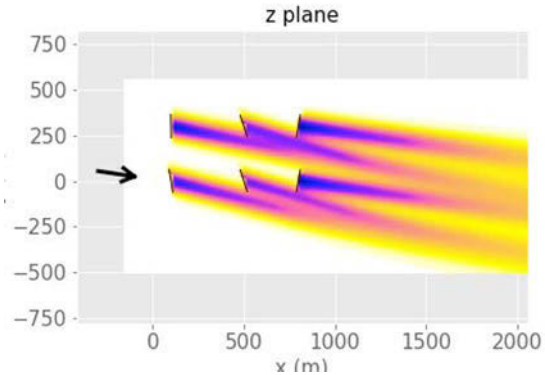


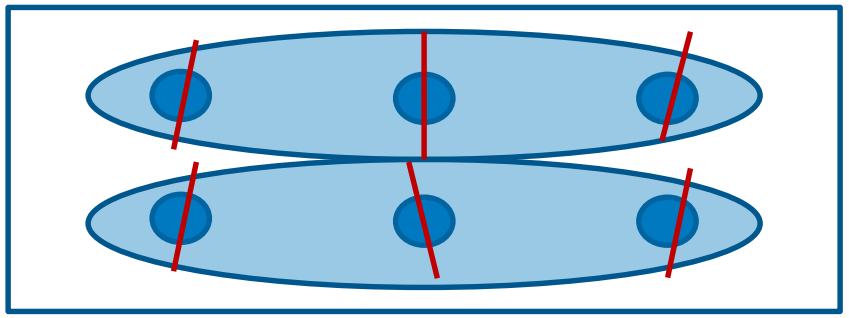
Tricking Floris



RL inherently plans over time. Much harder to do with explicit optimization when the model is nonlinear

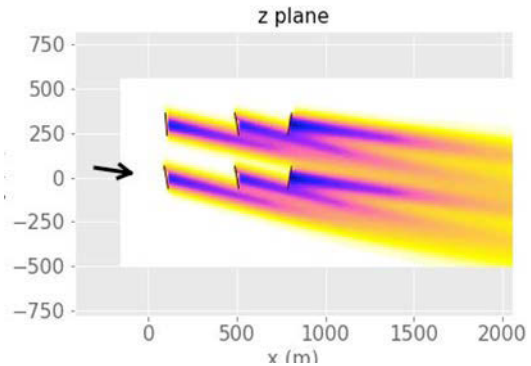
Wind farm 2: Distributed yaw control with ADMM-RL

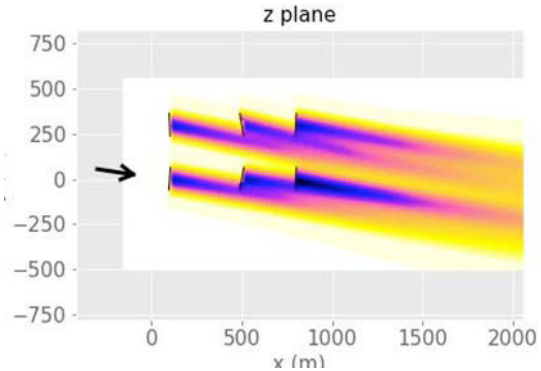


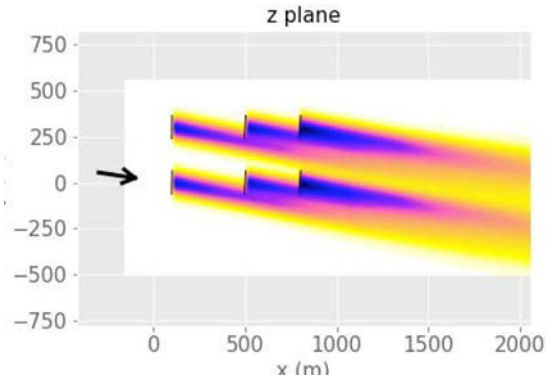


Floris unconstrained (81.6 Mw) (not realistic, but upper bound)

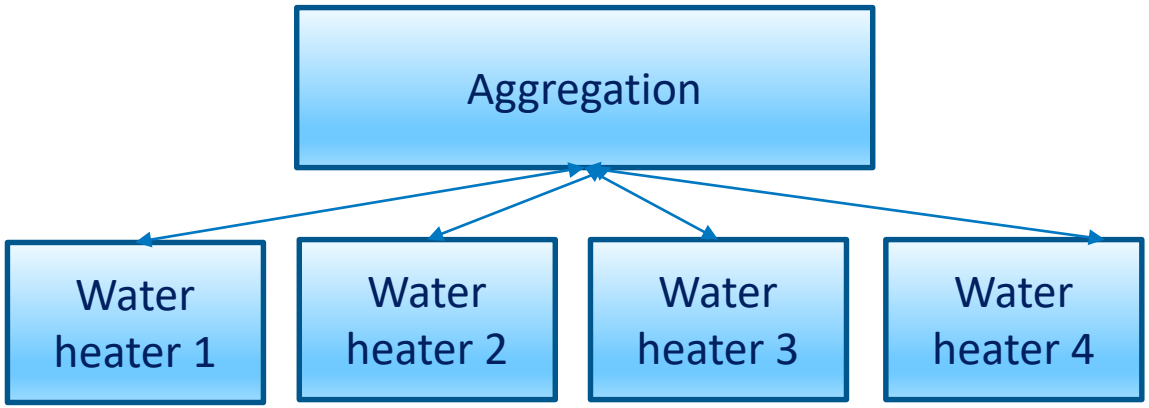
Floris constrained (79.0) **RL-central (**78.7) **ADMM-RL (**77.5)







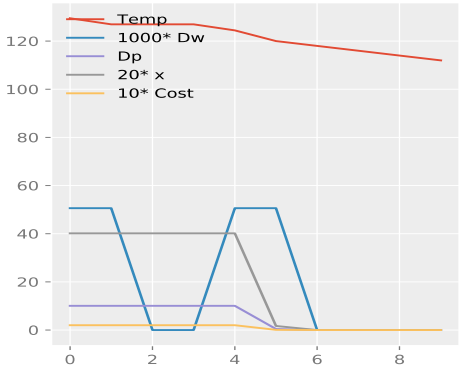
Demand response aggregation—a “sharing” problem



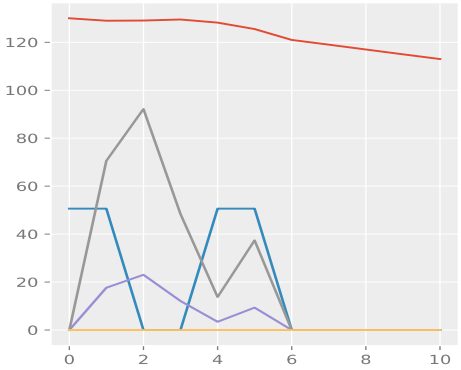
Objective: don’t use too much power

Objective: no cold showers

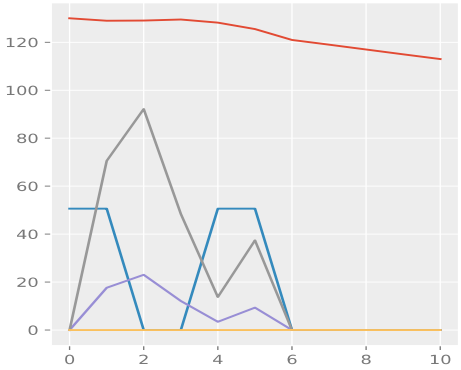
ADMM



ADMM-RL training



ADMM-operation



Parallel speedup provided by ADMM

Agent level speedup provided by learned RL controllers

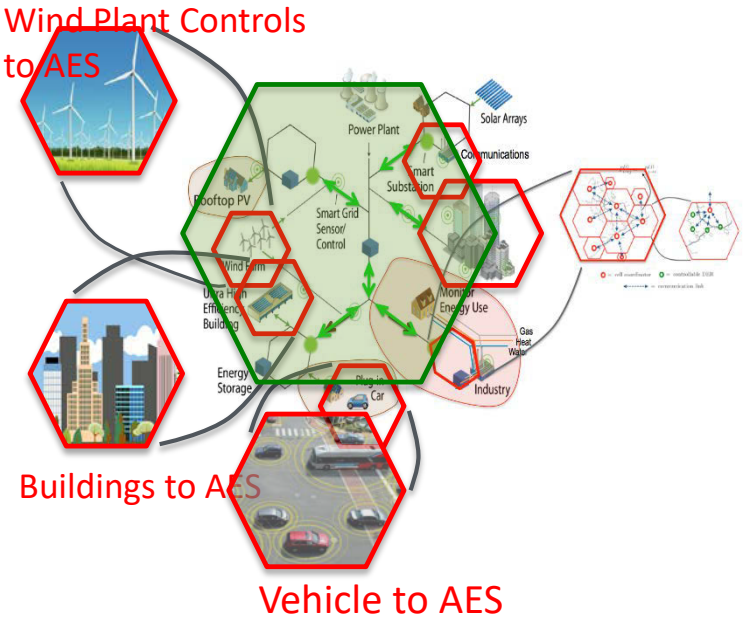
5x Speed up on mac, theoretical speedup is huge

Summary

- Energy systems rapidly becoming too complex to control optimally via real-time optimization.
- Reinforcement learning has potential to bypass online optimization and enable control of highly nonlinear stochastic systems.
- ADMM-RL extends RL to distributed control context.
- Proof of concept demonstrated on wind farm yaw control and demand response aggregation.

Swept under the rug / next steps

- **Every day is not the same! We have not addressed uncertainty and stochasticity**
- **What is the role of models? We should not just abandon what we already know.**
- **Tackling harder (sub-) problems with RL.**
- **More complex networks.**
- **Software infrastructure incorporating simulation, control, and learning.**



Thank you for having me!

Thank you

www.nrel.gov

NREL/PR-2C00-74798

This work was authored in part by the National Renewable Energy Laboratory (NREL), operated by Alliance for Sustainable Energy, LLC, for the U.S. Department of Energy (DOE) under Contract No. DE-AC36-08GO28308. This work was supported by the Laboratory Directed Research and Development (LDRD) Program at NREL. The views expressed in the article do not necessarily represent the views of the DOE or the U.S. Government. The U.S. Government retains and the publisher, by accepting the article for publication, acknowledges that the U.S. Government retains a nonexclusive, paid-up, irrevocable, worldwide license to publish or reproduce the published form of this work, or allow others to do so, for U.S. Government purposes.