Building an adjoint-based dynamics constrained optimization of electrical power systems.

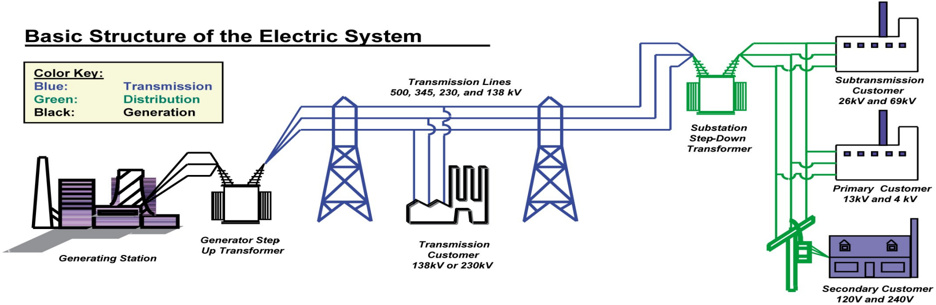
Paul Tranquilli[1]
Supervised by: Cosmin Petra and Shrirang Abhyankar

[1]Computational Science Laboratory (CSL)
Department of Computer Science
Virginia Tech

Outline

- Application: Electrical Power Grid
- Dynamics Constrained Optimal Power Flow
  - Mathematical formulation
  - Optimization requirements
- Obtaining DAE sensitivities
  - Finite Differences
  - Adjoint
- Implementation Considerations

Electrical Power System



Need **Economic** and **Secure** operation of power system.

- Current approach
  - Solve a nonlinear optimization problem commonly referred to as 'Optimal Power Flow (OPF)'
  - No information about system dynamics (state trajectories)
- We are incorporating the system dynamics leading to a DAE constrained optimization problem

Dynamics Constrained Optimal Power Flow

Mathematical Formulation

$$\begin{aligned} \min \quad & C(p) & \text{(Generation Cost)} \\ \text{s.t.} \quad & & \\ & g_s(p) = 0 & \text{(Power / current balance constraints)} \\ & h_s(p) \leq h^+ & \text{(Line flow constraints)} \\ & p^- \leq p \leq p^+ & \text{(Capacity and security constraints)} \\ & H(x(p,t), y(p,t)) \leq \rho & \text{(Dynamic constraint)} \end{aligned}$$

Where

$$H(x(p,t), y(p,t)) = \int_{t_0}^{t_F} \max(0, \omega - \omega_p, \omega_m - \omega)^\eta \, dt$$

is a cost functional coming from a differential-algebraic equation modeling the dynamics.

Solving the optimization problem

We make use of a nonlinear interior point optimization method which requires:

- The ability to evaluate the functions
  - $C(p)$, $g_s(p)$, $h_s(p)$,
  - and $H(x(p,t), y(p,t))$.
- As well as their derivatives
  - $\frac{\partial C}{\partial p}$, $\frac{\partial g_s}{\partial p}$, $\frac{\partial h_s}{\partial p}$,
  - and $\frac{\partial H}{\partial p}$.

This is straightforward for the algebraic constraints, not so easy for $H(x(p,t), y(p,t))$.

Computing $H(x(p,t), y(p,t))$

$$
\begin{array}{rcll}
M\dot{x} & = & f(t,x,y,p), & (x,y)|_{t=t_0} = (x_0(p), y_0(p)) \\
0 & = & g(t,x,y,p), & t_0 \leq t \leq t_F
\end{array}
$$

Where x and y are the differential and algebraic variables respectively, t is time, and p are the optimization variables. And

$$
g(t,x,y,p) = \begin{cases} g_1(t,x,y,p) & \text{if} \quad t_0 \leq t < t_f \\ g_2(t,x,y,p) & \text{if} \quad t_f \leq t < t_{cl} \\ g_1(t,x,y,p) & \text{if} \quad t_{cl} \leq t \leq t_F \end{cases}
$$

Solve for $x(t)$ and $y(t)$ using Crank-Nicholson time integration scheme, and compute the violation

$$
H(x(p,t), y(p,t)) = \int_{t_0}^{t_F} h(x(p,t), y(p,t))dt
$$

Computing the gradient of $H(x(p,t), y(p,t))$ I

Finite differences:

$$\frac{\partial H}{\partial p_i} = \frac{H(p_0 + \epsilon e_i) - H(p_0)}{\epsilon}$$

- Requires N solutions of the DAE, where N is the number of optimization variables p.
- Infeasible for even modestly sized problems.
- Extremely easy to implement.

Computing the gradient of $H(x(p,t), y(p,t))$ II

Adjoint sensitivities:

- Requires the solution of only a single, linear, backwards DAE.

$$\begin{array}{rcll} M^T \frac{d\lambda}{dt} & = & -f_x^T \lambda + g_x^T \mu - h_x & (\lambda, \mu)|_{t=t_F} = (\mathbf{0}, \mathbf{0}) \\ 0 & = & -f_y^T \lambda + g_y^T \mu - h_y & t_F \geq t \geq t_0 \end{array}$$

where the sensitivity is computed as

$$\frac{dH}{dp} = \int_{t_0}^{t_F} f_p^T \lambda \, dt - \left((M x_p)^T \lambda\right)\Big|_{t=t_0}$$

- Requires only two DAE solves, even as the size of the optimization parameter, p, grows.
- Requires a significant development investment, and expertise to derive and implement.

Implementing the adjoint

- Requires the use of a negative timestep. (Only current modification of PETSc)
- Requires forward solution, $(x(t), y(t))$, at each integration step.
- Extensive use of the User Contexts in PETSc allows for the freedom to implement adjoints, with minimal modification of the PETSc source code.

Finished and Ongoing Work

Completed

- Interfacing IPOPT with PETSc (Manually).
- Support for negative time-steps in PETSc.
- A working implementation of dynamics constrained optimization with finite difference sensitivities.
- Implementation of adjoints (Not directly in PETSc).

Ongoing

- Debugging adjoint implementation.

Future Work

- The construction of a rigorous mathematical framework for the derivation of adjoints for DAE systems with temporally discontinuous right hand sides.
- Simulating multiple faults(or contingency scenarios) requires further developments in the optimization and linear algebra (PIPS-NLP).
- A generalization of the current implementation to allow for different network topology, and larger problems.
- Integration with a mathematical modeling language, such as julia, for fast prototyping.

Questions?

Questions?