BLOM: Berkeley Library for Optimization Modeling

Sergey Vichik and Anthony Kelman

UC Berkeley
Department of Mechanical Engineering
Berkeley, CA

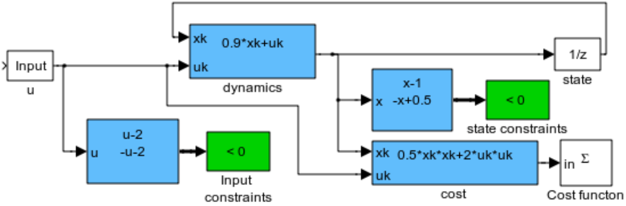
sergv@berkeley.edu,

March, 2012

What is BLOM ?

- A language of modeling dynamical nonlinear systems for optimization problems, especially MPC.
- Support for the following design phases:
  - Developing the model with an intuitive block diagram.
  - Forward simulation and validation of the model.
  - Automatic export of the optimization problem to a solver.
- Developed to handle non trivial problems
  - C++ or Matlab code generation.
  - Explicit evaluation of Jacobian and Hessian.
  - Proven with problems of tens of thousands variables.
- Eliminates manual problem coding, eases maintenance and assures that the same model used for optimization and for simulation.

"Hello World" example



$$\min_{u_k, x_k} \sum_k 0.5x_k^2 + 2u_k^2$$

$$\text{s.t.}: -2 \leqslant u_k \leqslant 2\,; 0.5 \leqslant x_k \leqslant 1\,; x_{k+1} = 0.9x_k + u_k$$

- The Functional block holds expression of the form $\frac{f(x)}{g(x)}$,
- The Constraint block marks variable as $\geqslant 0$ or $\leqslant 0$.
- The continuous or discrete State block.
- The Cost block, accumulates cost variables.
- The Input/External variable modifiers marks the control and the external variables.

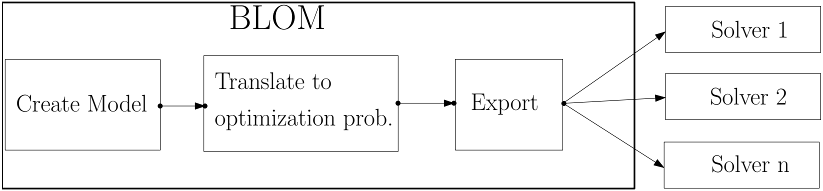
The functional block "Polyblock"

- Each polyblock is a polynomial-like function, that is described by two matrices, A and C. C holds the term coefficients and A defines the functions of variable to participate in the term.
- The polynomial-like function has the form: $f(x) = \sum_i \prod_j v_{i,j}(x_i)$. $v(x_i) \in \{{x^p}_{p\in\mathbb{R}}, \exp(x), \log(x)\}$.
- Example:

$$f(x) = 4x_1^3 + 0.2x_1^2x_2^{0.7} - 0.8x_1\exp(x_3) + 0.5\log(x_2)$$

$$c = \begin{bmatrix} c_1 \\ c_2 \\ c_3 \\ c_4 \end{bmatrix} = \begin{bmatrix} 4 \\ 0.2 \\ -0.8 \\ 0.5 \end{bmatrix} A = \begin{bmatrix} 3 & 0 & 0 \\ 2 & 0.7 & 0 \\ 1 & 0 & \inf \\ 0 & -\inf & 0 \end{bmatrix}.$$

BLOM work flow



- Create model using Simulink with BLOM library. Run and compare the model to a reference data.
- Translate to optimization problem: **ExtractModel(steps,dt,'RK4');**
- Export the problem to a solver: e.g. **CreateIpoptCPP**

BLOM status and features

1. Discrete and continuous models.
2. For continuous model, supports Euler, trapezoidal and RK4 discretization (easily expandable).
3. Full vector support.
4. Model developing features:
   - Color coded constraint violations.
   - Polyblocks display the user defined function.
   - User defined port labeling.
5. Export to IPOPT and fmincon solvers (more to come).
6. Used in joined project with UTRC for large HVAC MPC problem (dynamical model with 430 states, typically $\sim$ 30K variables in solver).

BLOM is fast

1. Explicit evaluation of the Jacobian and the Hessian.
2. Jacobian and Hessian are usually very sparse, and they are evaluated according to the sparsity pattern, therefore, only the non-zero elements are computed.
3. BLOM has an efficient C++ plug-in for IPOPT, that evaluates the cost, constraints and the aforementioned Jacobian and Hessian.
4. When used with IPOPT, the solver is a standalone executable, no additional interfacing overhead is added.
5. When used with IPOPT, the time of cost, constraints, Jacobian and Hessian evaluation is typically less than 10% of the total solver time.
6. The solver time is from milliseconds for small problems to minutes for tens of thousands variables sparse problems.