Implicit vs. explicit ODE

Fredrik Magnusson[1] **Karl Berntorp**[2]
Christan & Claus?

[1] Department of Automatic Control
Lund University, Sweden

[2] Mitsubishi Electric Research Laboratories
Cambridge, MA

October 15, 2015

Review

Last time:

- Compared simulation results for implicit DAE and explicit ODE with IDA and Radau5
- Implicit DAE formulation needs 10-20 times more steps
- Suppressing algebraic variables behaves very similarly to explicit ODE
- Pacejka Magic Formula (algebraic equation) stiff
- Surprisingly large relative errors with ODE formulation

Preview

This time:

- Revisit the error comparisons
- Discuss optimization formulation and solution procedure
- Discuss if the observed simulation phenomena regarding DAE vs. ODE are related to optimization

Partial DAE

- Last time considered eliminating all algebraics except the 4 nominal forces (Magic formula)
- I tested this as well, and call it DAE par.

Simulation results

Radau5 tolerance chosen to get reference solution, the rest to get similar accuracy

Setup	tol	Time [s]	Steps [1000]	Evals [1000]	Order
Radau5 DAE	1e-12	69	17	640	5
IDA DAE	1e-6	59	26	930	4
IDA DAE sup. alg.	1e-8	2.8	2.9	42	3
IDA DAE par.	1e-6	6.3	8	102	5
IDA ODE	1e-8	0.9	3.4	15	3

Radau5 DAE

```
Final Run Statistics: ---

 Number of steps                                 : 16701
 Number of function evaluations                  : 231074
 Number of Jacobian evaluations                  : 12338
 Number of function eval. due to Jacobian eval.  : 407154
 Number of error test failures                   : 1040
 Number of LU decompositions                     : 21266

Solver options:

 Solver                     : Radau5 (implicit)
 Tolerances (absolute)      : [  1.00000000e-12]
 Tolerances (relative)      : 1e-12
```

IDA DAE

```
Final Run Statistics: ---

 Number of steps                                       : 26361
 Number of function evaluations                        : 67823
 Number of Jacobian evaluations                        : 25991
 Number of function eval. due to Jacobian eval.        : 857703
 Number of error test failures                         : 12046
 Number of nonlinear iterations                        : 67823
 Number of nonlinear convergence failures              : 0

Solver options:

 Solver                       : IDA (BDF)
 Maximal order                : 5
 Suppressed algebr. variables : False
 Tolerances (absolute)        : 1e-06
 Tolerances (relative)        : 1e-06
```

IDA DAE sup. alg.

```
Final Run Statistics: ---

 Number of steps                                       : 2923
 Number of function evaluations                        : 7072
 Number of Jacobian evaluations                        : 1065
 Number of function eval. due to Jacobian eval.        : 35145
 Number of error test failures                         : 424
 Number of nonlinear iterations                        : 7072
 Number of nonlinear convergence failures              : 0

Solver options:

 Solver                       : IDA (BDF)
 Maximal order                : 5
 Suppressed algebr. variables : True
 Tolerances (absolute)        : 1e-08
 Tolerances (relative)        : 1e-08
```

IDA DAE par.

```
Final Run Statistics: ---

 Number of steps                                       : 8489
 Number of function evaluations                        : 18467
 Number of Jacobian evaluations                        : 5952
 Number of function eval. due to Jacobian eval.        : 83328
 Number of error test failures                         : 2867
 Number of nonlinear iterations                        : 18467
 Number of nonlinear convergence failures              : 0

Solver options:

 Solver                         : IDA (BDF)
 Maximal order                  : 5
 Suppressed algebr. variables   : False
 Tolerances (absolute)          : 1e-06
 Tolerances (relative)          : 1e-06
```

IDA ODE

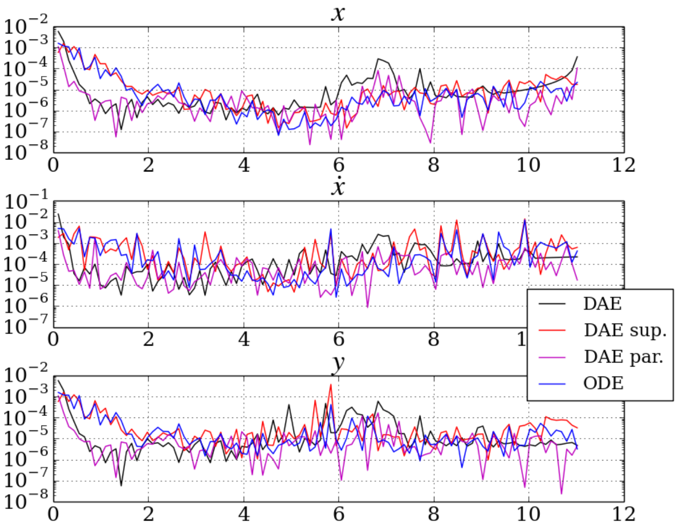
```
Final Run Statistics: ---

 Number of steps                                    : 3428
 Number of function evaluations                     : 5524
 Number of Jacobian evaluations                     : 919
 Number of function eval. due to Jacobian eval.     : 9190
 Number of error test failures                      : 477
 Number of nonlinear iterations                     : 5524
 Number of nonlinear convergence failures           : 0

Solver options:

 Solver                          : IDA (BDF)
 Maximal order                   : 5
 Suppressed algebr. variables    : False
 Tolerances (absolute)           : 1e-08
 Tolerances (relative)           : 1e-08
```

Error

- Reference solution v from Radau5 with `ATOL=RTOL=1e-12`
- Compare with the various IDA solutions $\hat{v}$, all with `ATOL=RTOL=1e-6`
- Compute relative error as function of time for each variable kind:

$$e_v(t) = \left\| \frac{v(t) - \hat{v}(t)}{v(t) + \epsilon_{\text{mach}}} \right\|_\infty, \quad \forall v \in \{\dot{x}, x, y\}$$

Variable kind errors



Variable kind errors

- Is uniformly distributed communication points sensible?
- Is constant order reasonable?
- Can we conclude that ODE formulation is superior for simulation for this case? Can we figure out why?

Commutativity

- Last time: Does BLT and fixed-step collocation commute?
- Consider single integration step of

$$\begin{aligned} \dot{x} &= f(x, y, u) \\ y &= g(x, u), \end{aligned} \tag{1}$$

and

$$\dot{x} = f(x, g(x, u), u) \tag{2}$$

Commutativity

Explicit DAE:

$$\dot{x}_k = f(x_k, y_k, u_k), \tag{3a}$$

$$y_k = g(x_k, u_k), \tag{3b}$$

$$\dot{x}_k = \frac{1}{h} \cdot \sum_{n=0}^{n_c} \alpha_{n,k} x_n, \tag{3c}$$

$$\forall k \in [1..n_c] \tag{3d}$$

Explicit ODE:

$$\dot{x}_k = f(x_k, g(x_k, u_k), u_k), \tag{4a}$$

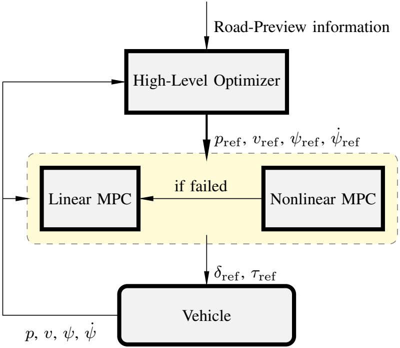
$$\dot{x}_k = \frac{1}{h} \cdot \sum_{n=0}^{n_c} \alpha_{n,k} x_n, \tag{4b}$$

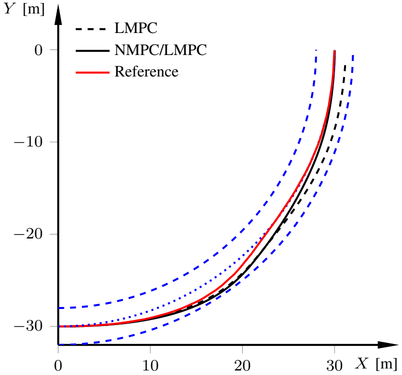
$$\forall k \in [1..n_c] \tag{4c}$$

Clearly commutative?

MPC

Bilevel control: Single-track used on high level to generate reference trajectory, double-track used on low level to follow trajectory





Convergence

- High level needs to be solved over free time horizon $\Longrightarrow$ difficult
- Low level has fixed time horizon (MPC) $\Longrightarrow$ less difficult
- Newton's method does not converge at all for high-level problem if we use full DAE, but converges for ODE
- Usually converges for low level even with DAE, but ODE more robust and faster

High level

Will today focus on high-level problem:

$$
\begin{aligned}
\underset{\delta_{\text{ref}},\tau_{f,\text{ref}},\tau_{r,\text{ref}}}{\text{minimize}} \quad & \int_{t_0}^{t_f} (\kappa_1 e^2 + \kappa_2 \beta^2)\,\mathrm{d}t \\
\text{subject to} \quad & |\tau_{i,\text{ref}}| \leq \tau_{i,\max},\ \forall i \in \{f, r\}, \\
& |\delta_{\text{ref}}| \leq \delta_{\max}, \\
& \|p(t_f) - p_{t_f}\| \leq \epsilon, \\
& \Gamma(p) \leq 0, \quad x(t_0) = x_0, \\
& F(\dot{x}, x, y, \delta_{\text{ref}}, \tau_{f,\text{ref}}, \tau_{r,\text{ref}}) = 0
\end{aligned}
\tag{5}
$$

Dynamic optimization

More abstract/compact formulation ($\boldsymbol{z} := (\dot{\boldsymbol{x}}, \boldsymbol{x}, \boldsymbol{y}, \boldsymbol{u})$):

$$
\begin{aligned}
&\text{minimize} && \int_{t_0}^{t_f} L(\boldsymbol{z}(t))\,\mathrm{d}t, && \text{(6a)}\\
&\text{with respect to} && \boldsymbol{x} : [t_0, t_f] \to \mathbb{R}^{n_x}, \quad \boldsymbol{y} : [t_0, t_f] \to \mathbb{R}^{n_y}, &&\\
& && \boldsymbol{u} : [t_0, t_f] \to \mathbb{R}^{n_u}, \quad t_f \in \mathbb{R}, &&\\
&\text{subject to} && \boldsymbol{F}(\boldsymbol{z}(t)) = \boldsymbol{0}, \qquad \boldsymbol{F}_0(\boldsymbol{z}(t_0)) = \boldsymbol{0}, && \text{(6b)}\\
& && \boldsymbol{z}_L \leq \boldsymbol{z}(t) \leq \boldsymbol{z}_U, && \text{(6c)}\\
& && \boldsymbol{g}(\boldsymbol{z}(t)) \leq \boldsymbol{0}, \qquad \boldsymbol{G}(\boldsymbol{z}(t_f)) \leq \boldsymbol{0}, && \text{(6d)}\\
& && \forall t \in [t_0, t_f]. &&
\end{aligned}
$$

NLP

Discretize differential equations to get a finite-dimensional nonlinear program (NLP) ($\boldsymbol{z}_{i,k} \approx \boldsymbol{z}(t_{i,k})$, where $t_{i,k}$ is collocation point k in element i):

$$\begin{aligned}
&\text{minimize} && \sum_{i=1}^{n_e} h_i \cdot (t_f - t_0) \cdot \sum_{k=1}^{n_c} \omega_k \cdot L\left(\boldsymbol{z}_{i,k}\right), && \text{(7a)}\\
&\text{with respect to} && \boldsymbol{z}_{i,k} \in \mathbb{R}^{2n_x+n_y+n_u}, \quad \boldsymbol{x}_{i,0} \in \mathbb{R}^{n_x}, \quad t_f \in \mathbb{R}, && \\
&\text{subject to} && \boldsymbol{F}(\boldsymbol{z}_{i,k}) = \boldsymbol{0}, \qquad \boldsymbol{F}_0(\boldsymbol{z}_{1,0}) = \boldsymbol{0}, && \text{(7b)}\\
& && \boldsymbol{u}_{1,0} = \sum_{k=1}^{n_c} \boldsymbol{u}_{1,k} \cdot \ell_k(0) \qquad \boldsymbol{z}_L \leq \boldsymbol{z}_{i,k} \leq \boldsymbol{z}_U, && \text{(7c)}\\
& && \boldsymbol{g}(\boldsymbol{z}_{i,k}) = \boldsymbol{0}, \qquad \boldsymbol{G}(\boldsymbol{z}_{n_e,n_c}) \leq \boldsymbol{0}, && \text{(7d)}\\
& && \forall (i,k) \in \{(1,0)\} \cup ([1..n_e] \times [1..n_c]), && \\
& && \dot{\boldsymbol{x}}_{j,l} = \frac{1}{h_j \cdot (t_f - t_0)} \cdot \sum_{m=0}^{n_c} \boldsymbol{x}_{j,m} \cdot \frac{\mathrm{d}\tilde{\ell}_m}{\mathrm{d}\tau}(\tau_l), && \text{(7e)}\\
& && \forall (j,l) \in [1..n_e] \times [1..n_c], && \text{(7f)}\\
& && \boldsymbol{x}_{n,n_c} = \boldsymbol{x}_{n+1,0}, \quad \forall n \in [1..n_e - 1]. && \text{(7g)}
\end{aligned}$$

NLP solution

After further abstraction, the NLP is:

$$
\begin{array}{ll}
\text{minimize} & f(x), \\
\text{with respect to} & x \in \mathbb{R}^m, \\
\text{subject to} & x_L \leq x \leq x_U, \\
 & g(x) = 0, \\
 & h(x) \leq 0.
\end{array}
$$

- Solved by IPOPT
- Lots of complicated details, but essentially Newton's method is applied on KKT optimality conditions
- See bonus slides for details

Convergence

Why convergence issues are more prominent in optimization than simulation in general:

- Larger system of equations (dual variables + TBVP)
- No good initial guess (at least not for high-level problem)
- Inherently ill-conditioned (see bonus slides)

Restoration phase

- For the high-level problem (if memory serves; to be confirmed by Karl), convergence is hampered by endless restoration
- Restoration is triggered for various reasons, but usually because of ill-conditioned Jacobian
- Restoration means that IPOPT stops solving the optimization problem and instead solves

$$
\begin{array}{ll}
\text{minimize} & ||g(x)||_1 + ||h(x) - y||_1 + 0.5\zeta||D_R(x - x_R)||_2^2, \\
\text{with respect to} & x \in \mathbb{R}^m, \quad y \in \mathbb{R}^n \\
\text{subject to} & x_L \leq x \leq x_U, \\
 & y \leq 0.
\end{array}
$$

- Failure to do this means that IPOPT can not even find a feasible point

Next steps

- Analyzing IPOPT convergence issues is complicated
  - Next step I see in that direction is to compare cond$(\nabla_z F)$ for DAE and ODE formulation
  - If significant difference, that is a likely explanation, but then we should try to figure out why there is a significant difference...
- Still don't understand why ODE is better for simulation
  - Probably not related to ill-conditioning?