# Numerical Simulation of Dynamic Systems: Hw10 - Solution

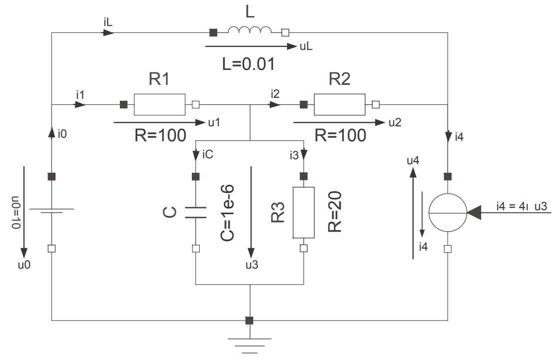
Prof. Dr. François E. Cellier
Department of Computer Science
ETH Zurich

May 14, 2013



## [H8.2] Inlining Radau IIA(3)

Given the electrical circuit:




- The circuit contains a constant voltage source, $u_0$, and a dependent current source, $i_4$, that depends on the voltage across the capacitor, $C$, and the resistor, $R_3$.
- Write down the element equations for the seven circuit elements. Since the voltage $u_3$ is common to two circuit elements, these equations contain 13 rather than 14 unknowns. Add the voltage equations for the three meshes and the current equations for three of the four nodes.

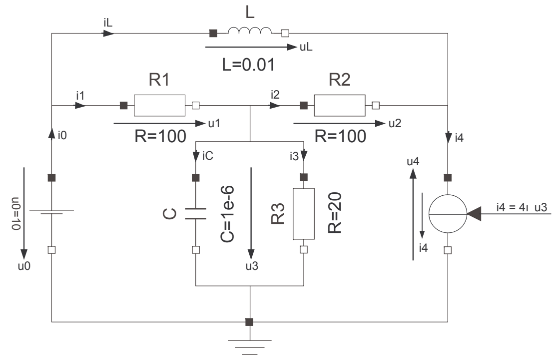


## [H8.2] Inlining Radau IIA(3) II

- We wish to inline the fixed-step 3$^{rd}$-order accurate Radau IIA algorithm. Draw the structure digraph of the inlined equation system, which now consists of 30 equations in 30 unknowns, and causalize it using the tearing method.
- Simulate the inlined difference equation system across $50\ \mu sec$ with zero initial conditions on both the capacitor and the inductor. Choose a step size of $h = 0.5\ \mu sec$. Use algebraic differentiation for the computation of the Hessian.
- Plot the voltage $u_3$ and the current $i_C$ on two separate subplots as functions of time.



## [H8.2] Inlining Radau IIA(3) III




1: $u_0 = 10$
2: $u_1 = R_1 \cdot i_1$
3: $u_2 = R_2 \cdot i_2$
4: $u_3 = R_3 \cdot i_3$
5: $i_C = C \cdot \frac{du_3}{dt}$
6: $u_L = L \cdot \frac{di_L}{dt}$
7: $i_4 = 4 \cdot u_3$

8: $u_0 = u_1 + u_3$
9: $u_L = u_1 + u_2$
10: $u_2 = u_3 + u_4$

11: $i_0 = i_1 + i_L$
12: $i_1 = i_2 + i_C + i_3$
13: $i_4 = i_2 + i_L$

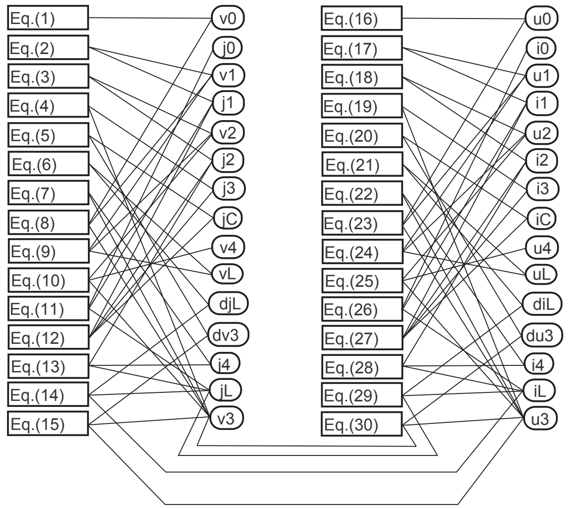
# [H8.2] Inlining Radau IIA(3) IV

$$\begin{array}{llcl}
1: & v_0 & = & 10 \\
2: & v_1 & = & R_1 \cdot j_1 \\
3: & v_2 & = & R_2 \cdot j_2 \\
4: & v_3 & = & R_3 \cdot j_3 \\
5: & j_C & = & C \cdot dv_3 \\
6: & v_L & = & L \cdot dj_L \\
7: & j_4 & = & 4 \cdot v_3 \\
\\
8: & v_0 & = & v_1 + v_3 \\
9: & v_L & = & v_1 + v_2 \\
10: & v_2 & = & v_3 + v_4 \\
\\
11: & j_0 & = & j_1 + j_L \\
12: & j_1 & = & j_2 + j_C + j_3 \\
13: & j_4 & = & j_2 + j_L \\
\\
14: & j_L & = & \mathrm{pre}(i_L) + \frac{5h}{12} \cdot dj_L - \frac{h}{12} \cdot di_L \\
15: & v_3 & = & \mathrm{pre}(u_3) + \frac{5h}{12} \cdot dv_3 - \frac{h}{12} \cdot du_3
\end{array}$$

$$\begin{array}{llcl}
16: & u_0 & = & 10 \\
17: & u_1 & = & R_1 \cdot i_1 \\
18: & u_2 & = & R_2 \cdot i_2 \\
19: & u_3 & = & R_3 \cdot i_3 \\
20: & i_C & = & C \cdot du_3 \\
21: & u_L & = & L \cdot di_L \\
22: & i_4 & = & 4 \cdot u_3 \\
\\
23: & u_0 & = & u_1 + u_3 \\
24: & u_L & = & u_1 + u_2 \\
25: & u_2 & = & u_3 + u_4 \\
\\
26: & i_0 & = & i_1 + i_L \\
27: & i_1 & = & i_2 + i_C + i_3 \\
28: & i_4 & = & i_2 + i_L \\
\\
29: & i_L & = & \mathrm{pre}(i_L) + \frac{3h}{4} \cdot dj_L + \frac{h}{4} \cdot di_L \\
30: & u_3 & = & \mathrm{pre}(u_3) + \frac{3h}{4} \cdot dv_3 + \frac{h}{4} \cdot du_3
\end{array}$$

Numerical Simulation of Dynamic Systems: Hw10 - Solution
Homework 10 - Solution
Inlining Radau IIA(3)

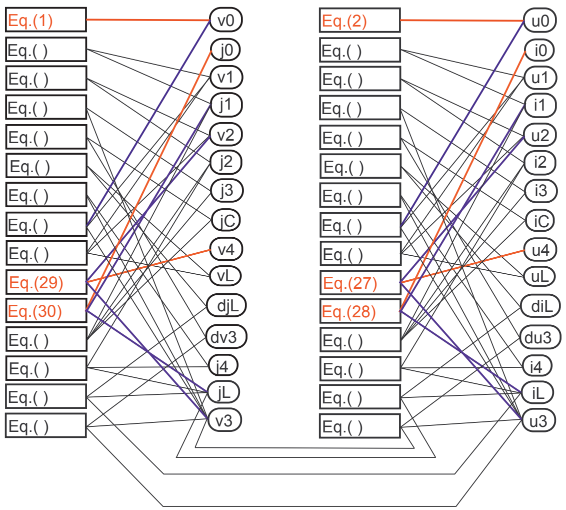
# [H8.2] Inlining Radau IIA(3) V




Numerical Simulation of Dynamic Systems: Hw10 - Solution
Homework 10 - Solution
Inlining Radau IIA(3)

# [H8.2] Inlining Radau IIA(3) VI

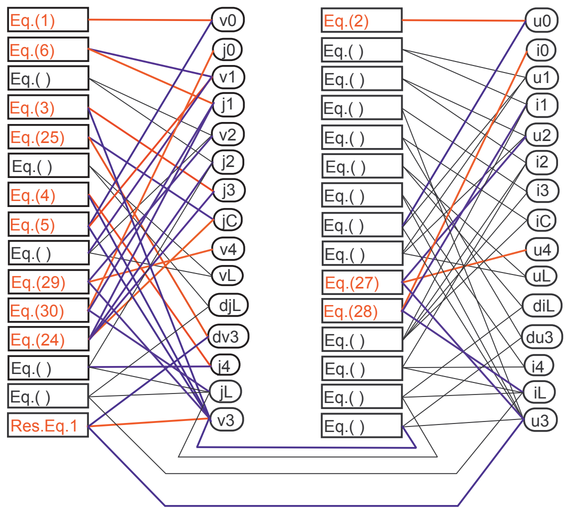
We can causalize 6 equations at once:




Numerical Simulation of Dynamic Systems: Hw10 - Solution
Homework 10 - Solution
Inlining Radau IIA(3)

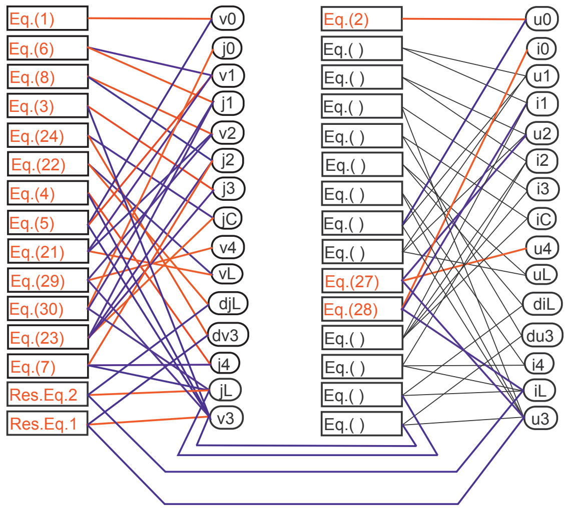
# [H8.2] Inlining Radau IIA(3) VII

We choose a $1^{st}$ tearing variable that allows us to causalize another 7 equations:



# [H8.2] Inlining Radau IIA(3) VIII

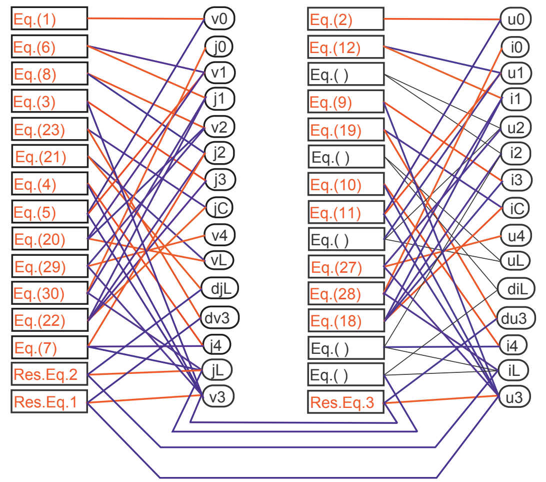
We choose a 2[nd] tearing variable that allows us to causalize another 5 equations:






# [H8.2] Inlining Radau IIA(3) IX

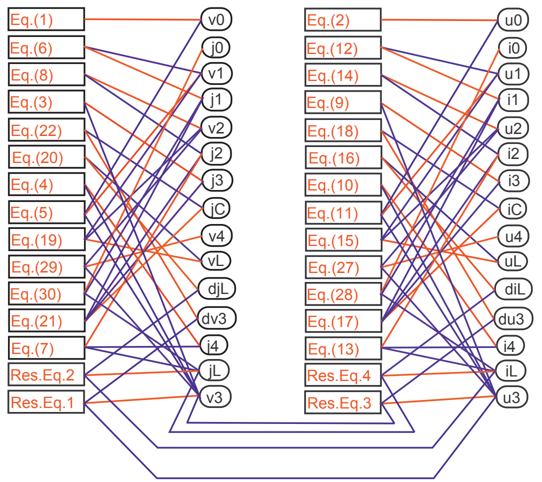
We choose a 3[rd] tearing variable that allows us to causalize another 7 equations:






# [H8.2] Inlining Radau IIA(3) X

We choose a 4[th] tearing variable that allows us to causalize the remaining 5 equations:






# [H8.2] Inlining Radau IIA(3) XI

$$
\begin{array}{llll}
1: & v_0 & = & 10 \\
2: & u_0 & = & 10 \\
3: & v_3 & = & R_3 \cdot j_3 \\
4: & j_4 & = & 4 \cdot v_3 \\
5: & v_0 & = & v_1 + v_3 \\
6: & v_1 & = & R_1 \cdot j_1 \\
7: & j_4 & = & j_2 + j_L \\
8: & v_2 & = & R_2 \cdot j_2 \\
9: & u_3 & = & R_3 \cdot i_3 \\
10: & i_4 & = & 4 \cdot u_3 \\
11: & u_0 & = & u_1 + u_3 \\
12: & u_1 & = & R_1 \cdot i_1 \\
13: & i_4 & = & i_2 + i_L \\
14: & u_2 & = & R_2 \cdot i_2 \\
15: & u_L & = & u_1 + u_2
\end{array}
$$

$$
\begin{array}{llll}
16: & u_L & = & L \cdot di_L \\
17: & i_1 & = & i_2 + i_C + i_3 \\
18: & i_C & = & C \cdot du_3 \\
19: & v_L & = & v_1 + v_2 \\
20: & v_L & = & L \cdot dj_L \\
21: & j_1 & = & j_2 + j_C + j_3 \\
22: & j_C & = & C \cdot dv_3 \\
23: & i_{L_{new}} & = & \mathrm{pre}(i_L) + \frac{3h}{4} \cdot dj_L + \frac{h}{4} \cdot di_L \\
24: & u_{3_{new}} & = & \mathrm{pre}(u_3) + \frac{3h}{4} \cdot dv_3 + \frac{h}{4} \cdot du_3 \\
25: & j_{L_{new}} & = & \mathrm{pre}(i_L) + \frac{5h}{12} \cdot dj_L - \frac{h}{12} \cdot di_L \\
26: & v_{3_{new}} & = & \mathrm{pre}(u_3) + \frac{5h}{12} \cdot dv_3 - \frac{h}{12} \cdot du_3 \\
27: & u_2 & = & u_3 + u_4 \\
28: & i_0 & = & i_1 + i_L \\
29: & v_2 & = & v_3 + v_4 \\
30: & j_0 & = & j_1 + j_L
\end{array}
$$

# [H8.2] Inlining Radau IIA(3) XII

| | | | |
|---|---|---|---|
| 1: | $v_0$ | $=$ | $10$ |
| 2: | $u_0$ | $=$ | $10$ |
| 3: | $j_3$ | $=$ | $\frac{1}{R_3}\cdot v_3$ |
| 4: | $j_4$ | $=$ | $4\cdot v_3$ |
| 5: | $v_1$ | $=$ | $v_0 - v_3$ |
| 6: | $j_1$ | $=$ | $\frac{1}{R_1}\cdot v_1$ |
| 7: | $j_2$ | $=$ | $j_4 - j_L$ |
| 8: | $v_2$ | $=$ | $R_2\cdot j_2$ |
| 9: | $i_3$ | $=$ | $\frac{1}{R_3}\cdot u_3$ |
| 10: | $i_4$ | $=$ | $4\cdot u_3$ |
| 11: | $u_1$ | $=$ | $u_0 - u_3$ |
| 12: | $i_1$ | $=$ | $\frac{1}{R_1}\cdot u_1$ |
| 13: | $i_2$ | $=$ | $i_4 - i_L$ |
| 14: | $u_2$ | $=$ | $R_2\cdot i_2$ |
| 15: | $u_L$ | $=$ | $u_1 + u_2$ |
| 16: | $di_L$ | $=$ | $\frac{1}{L}\cdot u_L$ |
| 17: | $i_C$ | $=$ | $i_1 - i_2 - i_3$ |
| 18: | $du_3$ | $=$ | $\frac{1}{C}\cdot i_C$ |
| 19: | $v_L$ | $=$ | $v_1 + v_2$ |
| 20: | $dj_L$ | $=$ | $\frac{1}{L}\cdot v_L$ |
| 21: | $j_C$ | $=$ | $j_1 - j_2 - j_3$ |
| 22: | $dv_3$ | $=$ | $\frac{1}{C}\cdot j_C$ |
| 23: | $i_{L_{new}}$ | $=$ | $\mathrm{pre}(i_L) + \frac{3h}{4}\cdot dj_L + \frac{h}{4}\cdot di_L$ |
| 24: | $u_{3_{new}}$ | $=$ | $\mathrm{pre}(u_3) + \frac{3h}{4}\cdot dv_3 + \frac{h}{4}\cdot du_3$ |
| 25: | $j_{L_{new}}$ | $=$ | $\mathrm{pre}(i_L) + \frac{5h}{12}\cdot dj_L - \frac{h}{12}\cdot di_L$ |
| 26: | $v_{3_{new}}$ | $=$ | $\mathrm{pre}(u_3) + \frac{5h}{12}\cdot dv_3 - \frac{h}{12}\cdot du_3$ |
| 27: | $u_4$ | $=$ | $u_2 - u_3$ |
| 28: | $i_0$ | $=$ | $i_1 + i_L$ |
| 29: | $v_4$ | $=$ | $v_2 - v_3$ |
| 30: | $j_0$ | $=$ | $j_1 + j_L$ |



# [H8.2] Inlining Radau IIA(3) XIII

We are now ready to code.

There are 4 tearing variables. Hence the Hessian matrix is of size $4 \times 4$.

*Algebraic differentiation* adds thus another $4 \cdot 30 = 120$ equations to the model.

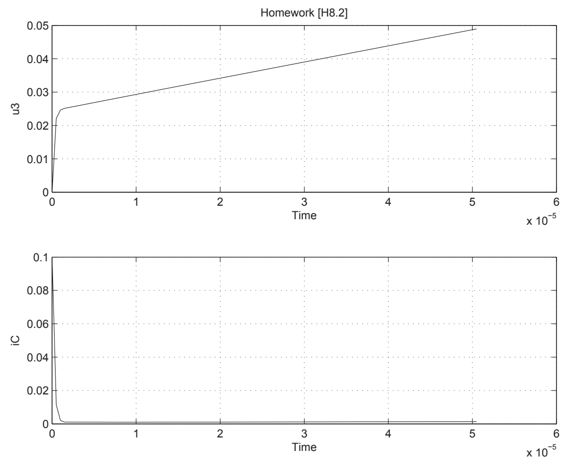
I coded a function radau_step that implements one step of **inlined Radau IIA(3)** applied to the circuit.

Since the problem has been inlined, the function radau_step contains both the model and the solver equations mixed together. It also includes the Newton iteration.

In my implementation, the function radau_step turned out to be 202 lines long.



# [H8.2] Inlining Radau IIA(3) XIV






# [H8.3] Step-size Control for Radau IIA(3)

We wish to augment the solution to problem [H8.2] by adding a step-size control algorithm.

- Use:

$$\mathbf{x}_{\mathbf{k+1}}^{\mathbf{blended}} = -\frac{1}{13}\,\mathbf{x}_{\mathbf{k-1}} + \frac{2}{13}\,\mathbf{x}_{\mathbf{k-\frac{2}{3}}} + \frac{14}{13}\,\mathbf{x}_{\mathbf{k}} - \frac{2}{13}\,\mathbf{x}_{\mathbf{k+\frac{1}{3}}} + \frac{11h}{13}\,\dot{\mathbf{x}}_{\mathbf{k+\frac{1}{3}}} + \frac{3h}{13}\,\dot{\mathbf{x}}_{\mathbf{k+1}}$$

as the embedding method for the purpose of error estimation, and use Fehlberg's step-size control algorithm:

$$h_{\text{new}} = \sqrt[5]{\frac{tol_{\text{rel}} \cdot \max(|x_1|, |x_2|, \delta)}{|x_1 - x_2|}} \cdot h_{\text{old}}$$

for the computation of the next step size. Of course, the formula needs to be slightly modified, since it assumes the error estimate to be $5^{th}$-order accurate, whereas in our algorithm, it is only $3^{rd}$-order accurate. Remember that the step size can never be modified two steps in a row.

# [H8.3] Step-size Control for Radau IIA(3) II

- Simulate the inlined difference equation system across 50 $\mu sec$ with zero initial conditions on both the capacitor and the inductor.
- Plot the voltage $u_3$, the current $i_C$, and the step size $h$ on three separate subplots as functions of time.



# [H8.3] Step-size Control for Radau IIA(3) III

I coded the step-size controlled algorithm as a cyclic method consisting of two semi-steps of half the step size. Both semi-steps make use of Radau IIA(3). After the two steps have been completed, the blended error method is being computed.

I coded the radau_step function as follows:

```
function [xnew, xtemp, xdotnew, xdottemp] = radau_step(x, t, h)
    ...
return
```

where xnew is the state vector at the end of the step, $\mathbf{x_{k+1}}$, xtemp is the intermediate state vector, $\mathbf{x_{k+\frac{1}{3}}}$, xdotnew is the state derivative vector at the end of the step, $\dot{\mathbf{x}}_{\mathbf{k+1}}$, and xdottemp is the intermediate derivative vector, $\dot{\mathbf{x}}_{\mathbf{k+\frac{1}{3}}}$.



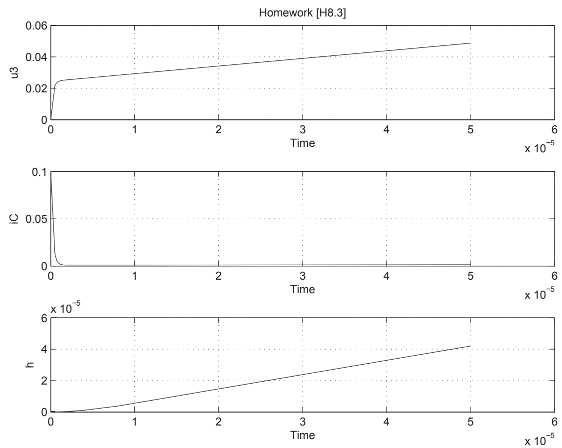
# [H8.3] Step-size Control for Radau IIA(3) IV

Step-size control can now be implemented as follows:

```
function [xnew, hnew] = radau_stepv(x, t, h)
    h2 = h/2;
    tol = 1e − 6;
    delta = 1e − 10;
    x1 = x;
    [x3, x2] = radau_step(x, t, h2);
    [xnew, x4, xnewdot, x4dot] = radau_step(x3, t + h2, h2);
    xblend = (−x1 + 2 ∗ x2 + 14 ∗ x3 − 2 ∗ x4 + (11 ∗ x4dot + 3 ∗ xnewdot) ∗ h2)/13;
    hnew = (tol∗max([norm(xnew),norm(xblend), delta])/norm(xnew − xblend)) ∧ (1/3) ∗ h;
return
```

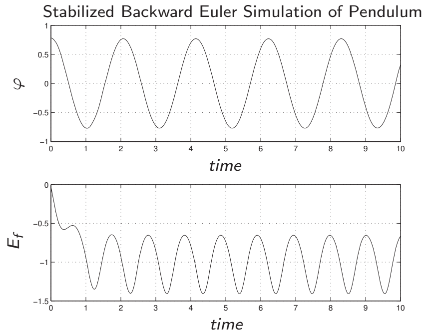


# [H8.3] Step-size Control for Radau IIA(3) V



# [H8.7] Stabilized BE Simulation of Overdetermined DAE System

In class (Presentation XX), I showed you how to stabilize the inlined BE simulation of the pendulum using an additional constraint equation. We obtained the following stabilized trajectories:






# [H8.7] Stabilized BE Simulation of Overdetermined DAE System II

On purpose, I haven't shown you the details of how these trajectories have been derived. In particular, I didn't provide you with a formula for when to end the Newton iteration. Since the linear system is now only solved in a least square sense, you can no longer test for $\|\mathcal{F}\|$ having decreased to a small value. The way I did it was to compute the norm of $\mathcal{F}$ and save that value between iterations. I then tested, whether the norm of $\mathcal{F}$ has no longer decreased significantly from one iteration to the next:

**while** abs($\|\mathcal{F}^{\ell}\| - \|\mathcal{F}^{\ell-1}\|$) $< 1.0e-6$,
  perform iteration
**end**,

Reproduce the graph showing the trajectories.



# [H8.7] Stabilized BE Simulation of Overdetermined DAE System III