10.34, Numerical Methods Applied to Chemical Engineering
Professor William H. Green
Lecture #15: Differential Algebraic Equations (DAEs). Introduction: Optimization.

Differential-Algebraic Equations (DAEs)

$\Im(\underline{x}(t), \dot{\underline{x}}(t)) = 0$ → ode15i

$\underline{\underline{M}}(\underline{x}) * \dot{\underline{x}} = \underline{G}(\underline{x})$ → ode15s

If $\underline{\underline{M}}$ is invertible, $\dot{\underline{x}} = \underline{\underline{M}}^{-1}(\underline{x})\underline{G}(\underline{x})$. ← ordinary ODE

Quasi-Steady-State Assumption (QSSA)

* make stiff equation into algebraic

$OH + CO \xrightarrow{k_1} H + CO_2$

$H + O_2 \xrightarrow{k_2} OH + O$

$O + H_2O \xrightarrow{k_3} 2OH$

$O \approx d[OH]/dt = k_2[H][O_2] - k_1[OH][CO] + 2k_3[O][H_2O]$

$O \approx d[H]/dt = k_1[OH][CO] - k_2[H][O_2]$

- originally had 6 Differential equations
- now have 2 algebraic and 4 differential equations
- takes you from ODE system to a DAE system

QSSA is not always helpful because ODE is faster and more accurate to solve. Solving a D.A.E. is like solving a stiff equation. QSSA has not removed original stiffness.

Another problem of DAE:

Consistent Initial Conditions

$\underline{x}(t_0), \dot{\underline{x}}(t_0) \qquad \Im(\underline{x}(t_0), \dot{\underline{x}}(t_0)) = 0$

You need both $\underline{x}$ and $\dot{\underline{x}}$. If $\dot{\underline{x}}$ is not provided, you have to use Newton's Method.

High index DAEs

DAE $\quad \Im = 0$ $\qquad$ Eventually you will have a normal ODE if you take enough derivatives.

↓ $\quad \dfrac{d\Im}{dt} = 0$ $\qquad$ High-index refers to system that requires a high degree of

ODE $\quad \dfrac{d^2\Im}{dt^2} = 0$ $\qquad$ differentiation. MATLAB stiff solvers or D. A. E. solvers cannot solve

these equations if the index is too high.

$$\frac{d\Im}{dt} = \sum \frac{\partial \Im}{\partial x_n}\frac{dx_n}{dt} + \sum \frac{\partial \Im}{\partial \dot{x}_n}\frac{d\dot{x}_n}{dt} = 0$$

$v_n \equiv \dot{x}_n \qquad \Im(\underline{x}, \underline{v}) = 0$

$$\sum \frac{\partial \Im}{\partial x_n} v_n + \underbrace{\sum \frac{\partial \Im}{\partial \dot{x}_n}}_{\underline{\underline{M}}(\underline{x},\underline{v})} \dot{v}_n = 0 \qquad M_{in} = \frac{\partial \Im_i}{\partial v_n}$$

Identity Matrix = $\underline{\underline{I}}$ 2n entries

$$\begin{pmatrix} \underline{\underline{M}} & \underline{\underline{O}} \\ \underline{\underline{O}} & \underline{\underline{I}} \end{pmatrix} \begin{pmatrix} \dot{\underline{v}} \\ \dot{\underline{x}} \end{pmatrix} = \begin{pmatrix} -\sum_n \frac{\partial \Im_i}{\partial x_n} v_n \\ \underline{v} \end{pmatrix} = \begin{pmatrix} -\sum_n \frac{\partial \Im_1}{\partial x_n} v_n \\ -\sum_n \frac{\partial \Im_2}{\partial x_n} v_n \\ \vdots \\ v_1 \\ v_2 \\ v_3 \end{pmatrix}$$

Eventually, matrix $\underline{\underline{M}}$ becomes invertible

If $\underline{\underline{M}}$ is invertible: $\dot{x} = \underline{\underline{M}}^{-1}(\underline{x})\underline{G}(\underline{x})$ ODE

Professor Barton is an expert in the field.

Predictor Corrector Method

$\underbrace{\underline{x}(t_{k-2}), \underline{x}(t_{k-1}), \underline{x}(t_k) \xrightarrow{\Delta t} \underline{x}(t_{k+1})}_{\text{polynomial fit}}$ Extrapolate → "Predictor" = initial guess for implicit solve

"Corrector": $\Im(\underline{x}_{k+1}, \underline{x}_{k+1}) = 0$

This method was developed by Bill Gear "Gear Predictor-Corrector".

"BDF polynomials" (Backward Differential Formula)→ guarantees numerical stability

Predictor is simple to find through extrapolation

Corrector is difficult and complicated

DASSL DASPK } Linda Petzold (Univ. California, Santa Cruz) Another similar to DASSL is DASAC

DASPK → Make Δt small in first few steps to minimize error

Another package is DASAC at the Univ. of Wisconsin, Madison. For linear equations, use MATLAB. Use small Δt on first step, because initial guess is low in information and equations take a long time to solve.

Optimization

$\min_{\underline{x}} f(x)$ with constraints $\underline{g}(\underline{x}) = 0$ and $\underline{h}(\underline{x}) \geq 0$.

What derivatives of f($\underline{x}$) can I compute easily?

1) None → simplex algorithms → fminsearch
2) Gradient $\underline{\nabla f}$
   a. Newton-type solvers
   b. Conjugate-gradient type solvers

Newton-type

$\underline{f}(\underline{x}+\underline{p}) = \underline{f}(\underline{x}) + \underline{\nabla} \underline{f}|_{\underline{x}} \cdot \underline{p} + \frac{1}{2}\underline{p}^T \underline{\underline{H}} \cdot \underline{p} + O(|\underline{p}|^3)$

$$\nabla \underline{f} = \begin{pmatrix} \frac{\partial f}{\partial x_1} \\ \frac{\partial f}{\partial x_2} \\ \frac{\partial f}{\partial x_3} \\ \vdots \end{pmatrix} \qquad \underline{\underline{H}} = \begin{pmatrix} \frac{\partial^2 f}{\partial x_1^2} & \frac{\partial^2 f}{\partial x_1 \partial x_2} & \frac{\partial^2 f}{\partial x_1 \partial x_3} & \cdots \\ \frac{\partial^2 f}{\partial x_2 \partial x_1} & \frac{\partial^2 f}{\partial x_2^2} & \frac{\partial^2 f}{\partial x_2 \partial x_3} & \cdots \\ & & & \end{pmatrix}$$

$\underline{f}(\underline{x}+\underline{p}) = \underline{f}(\underline{x}) + \nabla \underline{f}|_{\underline{x}} \cdot \underline{p} + \frac{1}{2}\underline{p}^T \underline{\underline{B}} \cdot \underline{p}$ → approximate Hessian; maybe use $\underline{\underline{I}}$

1) What direction is the next step? downhill
2) How far should I step? $\underline{f}(\underline{x}_{k+1}) < \underline{f}(\underline{x}_k)$

Contour Map

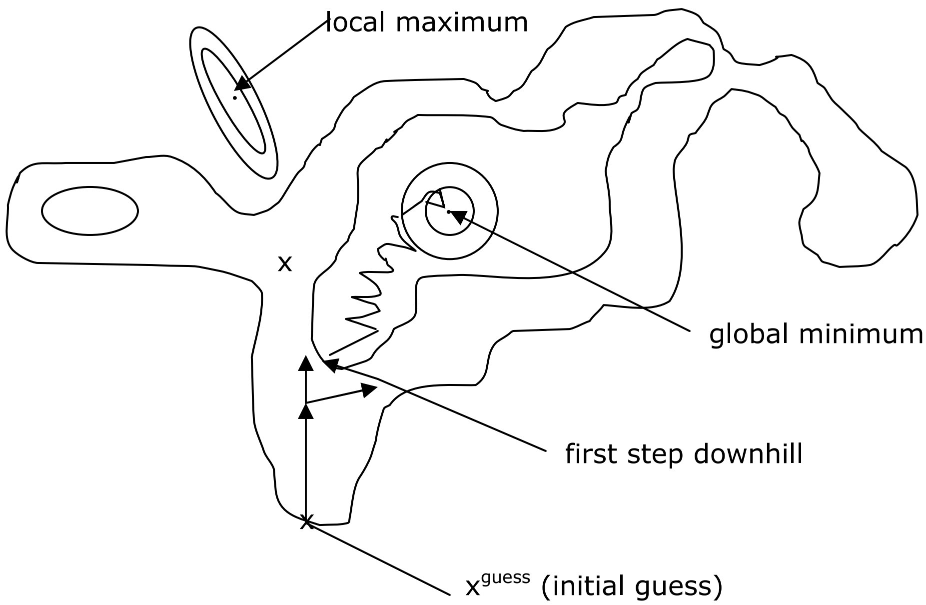

Figure 1. A contour map.

Choice #1: downhill

$$\underline{p} = c\left(-\frac{\nabla f}{\|\nabla f\|}\right)$$

optimal 'c', if $\underline{f}(\underline{x}+\underline{p}) = \underline{f}(\underline{x}) + \nabla \underline{f}|_{\underline{x}}\cdot\underline{p} + \frac{1}{2}\underline{p}^T\underline{\underline{B}}\cdot\underline{p}$ is true: Cauchy point

Choice #2: assume 2nd order expansion is exact

- Newton step
    - dangerous when you are far from the solution

Go downhill first to get closer and then apply Newton's Step.