Introduction to DAEs

Duy Duc NGUYEN

27/03/2019

Question 1: Overview

Let $\boldsymbol{F} : R^n \to R^n$, we search for solution $\boldsymbol{F}(\boldsymbol{x}) = \mathbf{0}$. It is a nonlinear algebraic equation system potentially very large: the number of unknowns and equations can be several million. Which methods can be used and compare advantages and dis-advantages of them.

Answer:

System of linear equations:

$$Ax = b$$

Square matrix: $A \in R^n \times R^n, x, b \in R^n,$

Symmetric: $AA^T = A^T A = I$

Krylov subspace: $K_k = span(r_{0,} Ar_o, A^2 r_o, \ldots, A^k r_o)$

basic matrix: $K_k = [v_{1,} v_{2,} \ldots, v_k],$ v_i is column *i*-th

Residual: $r = Ax - b$

System of non-linear equations:

$$F(x) = 0$$

Vector function $F : R^n \to R^n, x \in R^n$

Component: $F(x) = [f_i(x)], i = 1, \ldots, n, f_i : R^n \to R$

Jacobian matrix: $J(x) = F'(x) = [\partial f_i / \partial x_j(x)]$

Question 1: Newton method for solving non-liner system

ALGORITHM 5.3.1. newton(x, F, τ)

1. $r_0 = \|F(x)\|$
2. Do while $\|F(x)\| > \tau_r r_0 + \tau_a$
   (a) Compute $F'(x)$
   (b) Factor $F'(x) = LU$.
   (c) Solve $LUs = -F(x)$
   (d) $x = x + s$
   (e) Evaluate $F(x)$.

$\left.\begin{array}{l}\text{(b)}\\\text{(c)}\end{array}\right\} = s_n$

$$F(x) = 0,$$

$$F'(x_n) \approx \frac{F(x_{n+1}) - F(x_n)}{x_{n+1} - x_n}, \quad x_{n+1} = x_n + F'(x_n)^{-1}(F(x_{n+1}) - F(x_n)),$$

$$\|F(x_n)\| \gg \|F(x_{n+1})\| \rightarrow 0$$

So that $x_{n+1} = x_n - F'(x_n)^{-1} F(x_n)$

or $x_{n+1} - x_n = -F'(x_n)^{-1} F(x_n)$

Newton method:

$$\boxed{\begin{aligned} s_n &:= x_{n+1} - x_n \\ F'(x_n) s_n &= -F(x_n) \\ x_{n+1} &= x_n + s_n \end{aligned}}$$

Termination condition 1: $\|F(x)\| \le \tau_r \|F(x_0)\| + \tau_\alpha$

τ_r Relative error tolerance

τ_α Absolute error tolerance

Termination condition 2:
(Inexact Newton method)

$$\|F'(x_n) s_n + F(x_n)\| \le \eta_n \|F(x_n)\|$$

η_n forcing terms

C.T. Kelley – solving nonlinear equations with Newton's Method (1987)

Question 1: Newton-Krylov algorithms

Newton method

Advantage:
- Converging quickly to x*

Dis-advantage:
- Computing (step a) Jab matrix F' is costly
- Storing large Jab matrix F' is costly

Avoid F'

Krylov

GMRES | Bi-CGSTAB | TFQMR X

For solving linear system

Newton method + Krylov

Newton-GMRES | Newton-Bi-CGSTAB | Newton-TFQMR X

For solving non-linear system

Question 1: GMRES: Arnoldi

Description: produce an orthonormal basic of a space

Input: the first element of a basic of that space

Output:
$$K_k=[v_{1,}v_{2,}\dots,v_k],$$
$$\langle v_i,v_j\rangle=0, if\ i\neq j,$$
$$\|v_i\|=1$$

ALGORITHM 3.4.1. arnoldi(x_0, b, A, k, V)

1. Define $r_0 = b - Ax_0$ and $v_1 = r_0/\|r_0\|_2$.
2. For $i = 1, \dots, k-1$

$$v_{i+1} = \frac{Av_i - \sum_{j=1}^{i}((Av_i)^T v_j)v_j}{\|Av_i - \sum_{j=1}^{i}((Av_i)^T v_j)v_j\|_2}$$

Symmetric positive define (spd)

Orthogonal similarity transformation:

$$A=VHV^{-1}\rightarrow AV=VH$$

$$A_{n\times n}V_{n\times m}=V_{n\times m+1}H_{m+1\times m}$$

The last column *m*

$$Av_m=h_{1m}v_1+h_{2m}v_2+\dots+\boxed{h_{m+1,m}v_{m+1}}=\sum_{j=1}^{m}h_{jm}v_j$$

$$v_{m+1}=\frac{Av_m-\sum_{j=1}^{m}h_{jm}v_j}{h_{m+1,m}}$$

$$A=VHV^{-1}\rightarrow H=V^{-1}AV=\left(A^T\left(V^{-1}\right)^T\right)V=\left(AV\right)^TV$$
$$h_{ij}=[H]_{ij}=[Av_i]^Tv_j$$

C.T. Kelley – solving nonlinear equations with Newton's Method (1987)

Question 1: GMRES

ALGORITHM 3.4.2. gmresa$(x, b, A, \epsilon, kmax, \rho)$

1. $r = b - Ax$, $v_1 = r/\|r\|_2$, $\rho = \|r\|_2$, $\beta = \rho$, $k = 0$
2. While $\rho > \epsilon\|b\|_2$ and $k < kmax$ do
   (a) $k = k + 1$
   (b) for $j = 1, \dots, k$
   $h_{jk} = (Av_k)^T v_j$
   (c) $v_{k+1} = Av_k - \sum_{j=1}^{k} h_{jk} v_j$
   (d) $h_{k+1,k} = \|v_{k+1}\|_2$
   (e) $v_{k+1} = v_{k+1}/\|v_{k+1}\|_2$
   (f) $e_1 = (1, 0, \dots, 0)^T \in R^{k+1}$
   Minimize $\|\beta e_1 - H_k y^k\|_{R^{k+1}}$ over R^k to obtain y^k.
   (g) $\rho = \|\beta e_1 - H_k y^k\|_{R^{k+1}}$.
3. $x_k = x_0 + V_k y^k$.

arnoldi

Least square problem: Find x: $min_{x \in x_0 + K_k} \|r = b - Ax\|_2$

$$\forall z \in K_k, \exists y \in R^n : z = V_k y, V_k = [v_l^k]$$

$$x_0 + z = x_0 + V_k y$$

Least square problem: Find y $min_{y \in R^n} \|b - A(x_0 + V_k y)\|_2$

$$= min_{y \in R^n} \|r_0 - AV_k y\|_2, AV_k = V_k H,$$
$$= min_{y \in R^n} \|r_0 - V_k Hy\|_2$$

Termination of the iteration: $\|r_k\|_2 / \|b\|_2 \leq \epsilon$ and $k \leq kmax$

$$\beta = \|r_0\|_2, \quad e_1 = [1, 0, \dots, 0]$$

$$\beta V_k e_1 = \|r_0\|_2 v_1 = \|r_0\|_2 r_0 / \|r_0\|_2 = r_0$$

$$\|r_k\|_2 = \|r_0 - V_k Hy\|_2 = \|\beta V_k e_1 - V_k Hy\|_2$$
$$= \|V_k(\beta e_1 - Hy)\|_2 = \|\beta e_1 - Hy\|_{R^{k+1}}$$

(V is orthonormal basic matrix)

Description: find approximation of Ax=b
Input: $b, A, \epsilon, kmax, \rho$
Output: approximation of x^*

C.T. Kelley – solving nonlinear equations with Newton's Method (1987)

→ compute at each iteration: $AV_k y$

Question 1: Bi-CGSTAB

To understand this algorithm, must understand: CG, Bi-CG

CG: Conjugate gradient $x_{k+1}=x_k+\alpha_k p_k,\quad p_k \in K_k = span(r_0, Ar_0, A^2 r_0, \dots, A^k r_0)$

p_k search directions

$$r_{k+1}=b-Ax_{k+1}=b-A(x_k+\alpha_k p_k)=r_k-\alpha_k A p_k$$

orthogonality condition: $\langle r_{k+1}, r_k\rangle = \langle r_k - \alpha A p_k, r_k\rangle = 0$ So that $\boxed{\alpha_k = \frac{\langle r_k, r_k\rangle}{\langle Ap_k, r_k\rangle}}$

Assume that:

$p_{k+1}=r_{k+1}+\beta_k p_k \rightarrow r_k = p_k - \beta_k p_{k-1}$ Substitute into α get $\boxed{\beta_{k+1}=\frac{\langle r_{k+1}, r_{k+1}\rangle}{\langle r_k, r_k\rangle}}$

Bi-CG: Bi Conjugate gradient

Use two subspace: $K_k = span(r_{0,} Ar_o, A^2 r_o, \dots, A^k r_o)$ and $\bar{K}_k = span(\hat{r}_0, A^T \hat{r}_0, (A^T)^2 \hat{r}_0, \dots, (A^T)^{k-1} \hat{r}_0)$

$$r_{k+1}=r_k-\alpha_k A p_k,\quad \bar{r}_{k+1}=\bar{r}_k-\alpha_k A^T p_k$$

$$p_{k+1}=r_{k+1}+\beta_k p_k,\quad \bar{p}_{k+1}=\bar{r}_{k+1}+\beta_k \bar{p}_k$$

At each iteration $\langle \bar{r}_i, r_j\rangle = 0$, $\langle \bar{p}_i, Ap_j\rangle = 0$ if $i \neq j$

$$\boxed{\alpha_k = \frac{\langle r_k, \bar{r}_k\rangle}{\langle Ap_k, \bar{r}_k\rangle}}$$

$$\boxed{\beta_{k+1}=\frac{\langle r_{k+1}, \bar{r}_{k+1}\rangle}{\langle r_k, \bar{r}_k\rangle}}$$

Jonathan1994 - An introduction to CG method without the agonizing Pain

Vinay2016 – Understanding Bi-CGSTAB

C.T. Kelley – solving nonlinear equations with Newton's Method (1987)

Question 1: Bi-CGSTAB

CGS: Conjugate Gradient Squared

$$\text{if } p_0 = r_0 \rightarrow r_{k+1} = r_k - \alpha_k A p_k \rightarrow r_k = \phi_k(A) r_0 \qquad \bar{p}_0 = \bar{r}_0 \rightarrow \bar{r}_k = \phi_k(A^T) r_0$$

$$p_{k+1} = r_{k+1} + \beta_k p_k \rightarrow p_k = \pi_k(A) r_0 \qquad \bar{p}_k = \pi_k(A^T) \bar{r}_0$$

Relation $\phi_{k+1}(A) = \phi_k(A) - \alpha_k A \pi_k(A)$ And $\pi_{k+1}(A) = \phi_{k+1}(A) + \beta_k \pi_k(A)$

Bi-CGSTAB: Bi-conjugate gradient stabilized method

$$r_k = \phi_k(A) r_0 \text{ replaced by } r_k = \prod_{i=1}^{k} (I - w_i A) \phi_k(A) r_0 = \varphi_k(A) \phi_k(A) r_0 \qquad \bar{r}_k = \varphi_k(A^T) \phi_k(A^T) \bar{r}_0$$

$$p_k = \pi_k(A) r_0 \text{ replaced by } p_k = \prod_{i=1}^{k} (I - w_i A) \pi_k(A) r_0 = \varphi_k(A) \pi_k(A) r_0 \qquad \bar{p}_k = \varphi_k(A^T) \pi_k(A^T) \bar{r}_0$$

$$r_{k+1} = (I - w_i A) \varphi_k(A) \phi_{k+1}(A) r_0 = (I - w_i A) \varphi_k(A) [\phi_k(A) - \alpha_k A \pi_k(A)] r_0 = \ldots$$

$$\ldots = s_k - w_k A s_k, \text{ where } s_k = r_{k-1} - \alpha_k A p_k$$

$$p_{k+1} = (I - w_i A) \varphi_k(A) \pi_{k+1}(A) r_0 = (I - w_i A) \varphi_k(A) [\phi_{k+1}(A) + \beta_k \pi_k(A)] r_0 = \ldots$$

$$\ldots = r_k + \beta_k (p_k - w_k A p_k)$$

Question 1: Bi-CGSTAB

ALGORITHM 3.6.3. **bicgstab**$(x, b, A, \epsilon, kmax)$

1. $r = b - Ax$, $\hat{r}_0 = \hat{r} = r$, $\rho_0 = \alpha = \omega = 1$, $v = p = 0$, $k = 0$, $\rho_1 = \hat{r}_0^T r$
2. Do While $\|r\|_2 > \epsilon\|b\|_2$ and $k < kmax$
   (a) $k = k + 1$
   (b) $\beta = (\rho_k/\rho_{k-1})(\alpha/\omega)$
   (c) $p = r + \beta(p - \omega v)$ $\longleftarrow$ p_k
   (d) $v = Ap$
   (e) $\alpha = \rho_k/(\hat{r}_0^T v)$
   (f) $s = r - \alpha v$, $t = As$
   (g) $\omega = t^T s/\|t\|_2^2$, $\rho_{k+1} = -\omega\hat{r}_0^T t$
   (h) $x = x + \alpha p + \omega s$
   (i) $r = s - \omega t$ $\longleftarrow$ r_k

Compute β

$$\rho_k = \langle r_k, \bar{r}_k \rangle \quad \beta_{k+1} = \frac{\rho_{k+1}}{\rho_k} \frac{\alpha_k}{w_k}$$

Compute α

$$\rho_{1k} = \langle r_k, \bar{r}_0 \rangle \quad \alpha_k = \frac{\rho_{1k}}{\langle Ap_k, \bar{r}_0 \rangle} = \frac{r_k^T \bar{r}_0}{\bar{r}^T{}_0 A p_k}$$

Compute w

$$w_k = \frac{\langle As_k, s_k \rangle}{\langle As_k, As_k \rangle} = \frac{(t_k)^T s_k}{\|t_k\|^2}$$

Return approximate solution

$$x_{k+1} = x_k + \alpha_k p_k + w_k s_k$$

Vort1992 – Bi-CGSTAB a fast and smoothly converging variant of Bi-CG for the solution of nonsymetric linear systems

Jonathan1994 - An introduction to CG method without the agonizing Pain

Vinay2016 – Understanding Bi-CGSTAB

C.T. Kelley – solving nonlinear equations with Newton's Method (1987)

Question 1: Compare GMRES and Bi-CGSTAB

Only need matrix-vector products

GMRES

At iteration m ≤ kmax

Store $x, F(x), x+hv, F(x+hv), s, \{v_k\}_{k=1}^{m}$

Require new storage at m+1

If run out of storage, then restart with $x_0 := x_k$

Slow down the speed of convergence

Bi-CGSTAB

At iteration m ≤ kmax

Store $x, b, r, p, v, \bar{r}_0, t$

No new storage required

I choose this one

Question 1: Newton-GMRES

$$F'(x_n)\, s_n = -F(x_n)$$

F'(x) considered as A, -F(x) considered as b, r_k approximates b, r := -F(x)

Algorithm 6.2.1. $\mathtt{fdgmres}(s, x, F, h, \eta, kmax, \rho)$

1. $\boxed{s = 0,\ r = -F(x),}$ $v_1 = r/\|r\|_2$, $\rho = \|r\|_2$, $\beta = \rho$, $k = 0$
2. While $\rho > \eta\|F(x)\|_2$ and $k < kmax$ do
   (a) $k = k + 1$
   (b) $v_{k+1} = D_h F(x : v_k)$
   for $j = 1, \ldots k$
   i. $h_{jk} = v_{k+1}^T v_j$
   ii. $v_{k+1} = v_{k+1} - h_{jk} v_j$
   (c) $h_{k+1,k} = \|v_{k+1}\|_2$
   (d) $v_{k+1} = v_{k+1}/\|v_{k+1}\|_2$
   (e) $e_1 = (1, 0, \ldots, 0)^T \in R^{k+1}$
   Minimize $\|\beta e_1 - H_k y^k\|_{R^{k+1}}$ to obtain $y^k \in R^k$.
   (f) $\rho = \|\beta e_1 - H_k y^k\|_{R^{k+1}}$.
3. $s = V_k y^k$. **Output** s_n

arnoldi

Termination condition 2

Forward difference GMRES algorithm works with F instead of (A,b)

Algorithm 6.3.1. $\mathtt{nsolgm}(x, F, \tau, \eta)$

1. $r_c = r_0 = \|F(x)\|_2/\sqrt{N}$
2. Do while $\|F(x)\|_2/\sqrt{N} > \tau_r r_0 + \tau_a$
   (a) Select η.
   (b) $\mathtt{fdgmres}(s, x, F, \eta)$ $= s_n$
   (c) $x = x + s$
   (d) Evaluate $F(x)$
   (e) $r_+ = \|F(x)\|_2/\sqrt{N}$, $\sigma = r_+/r_c$, $r_c = r_+$
   (f) If $\|F(x)\|_2 \le \tau_r r_0 + \tau_a$ exit.

Termination condition 1

Algorithm computes for a better eta

Forward difference approximation of (eq5.15)

$$F'(x)\, v_k$$

approximate multiplication of F' with a vector, mainly by working with F, instead of compute F'

C.T. Kelley – solving nonlinear equations with Newton's Method (1987)

Question 1: Newton-BiCGSTAB

ALGORITHM 3.6.3. **bicgstab**$(x, b, A, \epsilon, kmax)$

1. $\cancel{r = b - Ax}$, $\hat{r}_0 = \hat{r} = r$, $\rho_0 = \alpha = \omega = 1$, $v = p = 0$, $k = 0$, $\rho_1 = \hat{r}^T r$

   S=0, r= -F(x)

2. Do While $\|r\|_2 > \epsilon \|b\|_2$ and $k < kmax$
   (a) $k = k + 1$
   (b) $\beta = (\rho_k / \rho_{k-1})(\alpha / \omega)$
   (c) $p = r + \beta(p - \omega v)$
   (d) $v = Ap$ $\quad v_k = D_h F(x : p_k)$
   (e) $\alpha = \rho_k / (\hat{r}_0^T v)$
   (f) $s = r - \alpha v$, $t = As$ $\quad t_k = D_h F(x : s_k)$
   (g) $\omega = t^T s / \|t\|_2^2$, $\rho_{k+1} = -\omega \hat{r}_0^T t$
   (h) $x = x + \alpha p + \omega s$ $\quad S_{k+1} = S_k + \alpha_k p_k + w_k s_k$
   (i) $r = s - \omega t$

ALGORITHM 6.3.1. **nsolgm**(x, F, τ, η)

1. $r_c = r_0 = \|F(x)\|_2 / \sqrt{N}$
2. Do while $\|F(x)\|_2 / \sqrt{N} > \tau_r r_0 + \tau_a$
   (a) Select η.
   (b) ~~fdgmres~~ (s, x, F, η) dfbicgstab
   (c) $x = x + s$
   (d) Evaluate $F(x)$
   (e) $r_+ = \|F(x)\|_2 / \sqrt{N}$, $\sigma = r_+ / r_c$, $r_c = r_+$
   (f) If $\|F(x)\|_2 \le \tau_r r_0 + \tau_a$ exit.

C.T. Kelley – solving nonlinear equations with Newton's Method (1987)

Question 2: Overview

Question 2: Consider $\boldsymbol{F} : R^n \to R^n$, a system of differential and algebraic equations (DAEs):

$$\boldsymbol{F}(\boldsymbol{x}(t), t) = \boldsymbol{I}(\boldsymbol{x}(t)) + \dot{\boldsymbol{Q}}(\boldsymbol{x}(t)) + \boldsymbol{S}(t)$$

Where $\boldsymbol{x}(t)$ is the solution of $\boldsymbol{F}(\boldsymbol{x}(t), t) = 0$ for $(t \in [0, T])$. $\boldsymbol{I}$ and $\boldsymbol{Q}$ are algebraic nonlinear functions of x and $\boldsymbol{S}$ is an explicit term function of t (stimuli of the system).

Additional informations:

- $\dot{\boldsymbol{Q}}$ means time derivative of $\boldsymbol{Q}$.
- $\boldsymbol{F}, x, I, Q$ and $S \in R^n$,
- n the size of the system (number of equations in $\boldsymbol{F}$, or number of unknowns in $\boldsymbol{x}$) can be very big (up to several millions)
- $\boldsymbol{I}$ and $\boldsymbol{Q}$ are $\boldsymbol{C}^1$ functions (differentiable with continuous derivatives).
- We consider that the system has a solution that is unique
- $\boldsymbol{F}$ is a stiff system (large ratio in time constants)

The question: Propose a method to compute a numerical approximation of the solution $x(t)$ of this system $F(x(t), t) = 0$ for $(t \in [0, T])$.

$$\frac{d\;\mathbf{Q}(\mathbf{x}(t))}{dt}=\frac{d\mathbf{Q}}{d\,\mathbf{x}}\frac{d\,\mathbf{x}}{dt}=Jab(Q,x)\,\dot{\mathbf{x}}$$

system becomes

$$\boldsymbol{I}(\mathbf{x}(t))+Jab(\mathbf{Q},\mathbf{x})\dot{\mathbf{x}}+\boldsymbol{S}(\mathbf{x})=\boldsymbol{F}(t,\mathbf{x},\dot{\mathbf{x}})=\mathbf{0}$$

Forward Euler: $\boldsymbol{F}\left(t_{n-1},\mathbf{x}_{n-1},\frac{\mathbf{x}_n-\mathbf{x}_{n-1}}{\Delta t_n}\right)=\mathbf{0}$ → Cfl condition, 1st order convergence

Backward Euler: $\boldsymbol{F}\left(t_n,\mathbf{x}_n,\frac{\mathbf{x}_n-\mathbf{x}_{n-1}}{\Delta t_n}\right)=\mathbf{0}$ → Stable, 1st order convergence → Newton-krylov method

Mid-point: $\boldsymbol{F}\left(\frac{t_n+t_{n-1}}{2},\frac{\mathbf{x}_n+\mathbf{x}_{n-1}}{2},\frac{\mathbf{x}_n-\mathbf{x}_{n-1}}{\Delta t_n}\right)=\mathbf{0}$ → ½ Stable, 2nd order convergence

K.E. Brenan, S.L. Campbell, R.L. Petzold Numerical Solution of Initial Value Problems in DAEs

Thank you for your attention