Chap. 4-4. Flux Functions for 2-D Euler Equations

- ***FVS Methods***
  - $\mathbf{F} = \mathbf{F}^+ + \mathbf{F}^-$ such that $\mathbf{F}^\pm$ has positive/negative eigenvalues $\rightarrow \mathbf{F}_{i+1/2,j} = \mathbf{F}^+_{i,j} + \mathbf{F}^-_{i+1,j}$
    - S-W FVS

• $\mathbf{F} = A\mathbf{U} \;\rightarrow\; \mathbf{F}^\pm = A^\pm \mathbf{U}$ with $A^\pm = R\Lambda^\pm R^{-1}$

$$\mathbf{F}^\pm = A^\pm \mathbf{U} = R\Lambda^\pm R^{-1}\mathbf{U} = \frac{\rho}{2\gamma}\mathbf{r}_1\lambda_1^\pm + \frac{(\gamma-1)}{\gamma}\rho\mathbf{r}_2\lambda_2^\pm + \frac{\rho}{2\gamma}\mathbf{r}_4\lambda_4^\pm = \frac{\rho}{2\gamma}\begin{bmatrix} \lambda_1^\pm + 2(\gamma-1)\lambda_2^\pm + \lambda_4^\pm \\ (u-c)\lambda_1^\pm + 2(\gamma-1)u\lambda_2^\pm + (u+c)\lambda_4^\pm \\ v\lambda_1^\pm + 2(\gamma-1)v\lambda_2^\pm + v\lambda_4^\pm \\ (H-uc)\lambda_1^\pm + (\gamma-1)(u^2+v^2)\lambda_2^\pm + (H+uc)\lambda_4^\pm \end{bmatrix}$$

From $\hat{\mathbf{F}} = \mathbf{F}(\hat{\mathbf{U}}) = \hat{A}\hat{\mathbf{U}}$ and $\hat{\mathbf{F}}^\pm = \hat{A}^\pm\hat{\mathbf{U}}$, $\hat{\mathbf{F}}^\pm = \mathbf{F}^\pm$ with $\begin{pmatrix} u \rightarrow U \\ v \rightarrow V \end{pmatrix}$

- van Leer FVS

$$\bullet\ \mathbf{F} = \begin{bmatrix} \rho u \\ \rho u^2 + p \\ \rho uv \\ \rho uH \end{bmatrix} = \begin{bmatrix} \rho c M \\ \rho c^2\left(\gamma M^2 + 1\right)/\gamma \\ \rho c M v \\ \rho c^3 M\left(0.5M^2 + 1/(\gamma-1)\right) + 0.5\rho c M v^2 \end{bmatrix} \;\rightarrow\; \mathbf{F}^\pm = \pm\frac{\rho c}{4}(1\pm M)^2 \begin{bmatrix} 1 \\ \dfrac{2c}{\gamma}\left(\dfrac{\gamma-1}{2}M \pm 1\right) \\ v \\ \dfrac{2c^2}{\gamma^2-1}\left(\dfrac{\gamma-1}{2}M \pm 1\right)^2 + \dfrac{v^2}{2} \end{bmatrix}$$

Thus, $\hat{\mathbf{F}}^\pm$ with $M \rightarrow \dfrac{U}{c}$ and $v \rightarrow V$

- ***FDS Methods***
  - From $\Delta\hat{\mathbf{F}} = \tilde{A}\hat{\mathbf{U}}$, split $\Delta\hat{\mathbf{F}}^{\pm} = \tilde{A}^{\pm}\Delta\hat{\mathbf{U}}$ with $\tilde{A}^{\pm} = \tilde{R}\tilde{\Lambda}^{\pm}\tilde{R}^{-1}$
  - $$\tilde{A}_{i+1/2,j} = \begin{bmatrix} 0 & 1 & 0 & 0 \\ (\gamma-1)\tilde{H} - \left(\tilde{U}^2 + \tilde{c}^2\right) & (3-\gamma)\tilde{U} & (1-\gamma)\tilde{V} & (\gamma-1) \\ -\tilde{U}\tilde{V} & \tilde{V} & \tilde{U} & 0 \\ \frac{\tilde{V}}{2}\left[(\gamma-3)\tilde{H} - \tilde{c}^2\right] & \tilde{H} - (\gamma-1)\tilde{U}^2 & (1-\gamma)\tilde{U}\tilde{V} & \gamma\tilde{U} \end{bmatrix}$$

    with $\tilde{\rho} = \sqrt{\rho_{i,j}\rho_{i+1,j}}$, $\tilde{U} = \dfrac{\sqrt{\rho_{i,j}}U_{i,j} + \sqrt{\rho_{i+1,j}}U_{i+1,j}}{\sqrt{\rho_{i,j}} + \sqrt{\rho_{i+1,j}}}$ (and $V_{i,j}, V_{i+1,j}$ for $\tilde{V}$)

    $$\tilde{H} = \frac{\sqrt{\rho_{i,j}}H_{i,j} + \sqrt{\rho_{i+1,j}}H_{i+1,j}}{\sqrt{\rho_{i,j}} + \sqrt{\rho_{i+1,j}}}, \tilde{c} = \sqrt{(\gamma-1)\left[\tilde{H} - \frac{\tilde{U}^2 + \tilde{V}^2}{2}\right]}$$
- **Roe**
  - $\hat{\mathbf{F}}_{i+1/2,j} = \dfrac{1}{2}\left(\hat{\mathbf{F}}_{i,j} + \hat{\mathbf{F}}_{i+1,j}\right) - \dfrac{1}{2}\left|\tilde{A}_{i+1/2,j}\right|\Delta\hat{\mathbf{U}}_{i+1/2,j}$
- **RoeM**
  - Mach number-based weighting functions *(f, g)* are the same as before.

    $$\hat{\mathbf{F}}_{i+\frac{1}{2},j} = \frac{b_1 \times \hat{\mathbf{F}}_{i,j} - b_2 \times \hat{\mathbf{F}}_{i+1,j}}{b_1 - b_2} + \frac{b_1 \times b_2}{b_1 - b_2}\Delta\hat{\mathbf{U}}^*_{i+1/2,j} - g\frac{b_1 \times b_2}{b_1 - b_2} \times \frac{1}{1+|\tilde{M}|}\mathbf{B}\Delta\hat{\mathbf{U}}_{i+1/2,j}$$

Chap. 4-4. Flux Functions for 2-D Euler Equations

- ***Hybrid Flux Splitting Methods***

- $$\mathbf{F} = \mathbf{F}_c + \mathbf{F}_p = u\begin{bmatrix} \rho \\ \rho u \\ \rho v \\ \rho H \end{bmatrix} + \begin{bmatrix} 0 \\ p \\ 0 \\ 0 \end{bmatrix} = u\mathbf{w} + \begin{bmatrix} 0 \\ p \\ 0 \\ 0 \end{bmatrix}$$

- **AUSM/AUSM+**

$$\mathbf{F}_{i+1/2,j} = \begin{bmatrix} M_{i+1/2,j}(c\mathbf{w}_c)_{i+1/2,j} \\ M_{i+1/2,j}c_{i+1/2,j}\mathbf{w}_{i+1/2,j} \end{bmatrix} + \begin{bmatrix} 0 \\ p_{i+1/2,j} \\ 0 \\ 0 \end{bmatrix} \text{ with } M_{i+1/2,j} = M^+_{i,j} + M^-_{i+1,j} \text{ and } M_{i,j} = \begin{cases} u_{i,j}/c_{i,j} \\ \text{or} \\ u_{i,j}/c_{i+1/2,j} \end{cases}$$

Thus, $\hat{\mathbf{F}}_{i+1/2,j}$ by $u \to U,\ v \to V$ and $c_{i+1/2,j} = \min\left(\tilde{c}_{i,j}, \tilde{c}_{i+1,j}\right), \tilde{c} = c^{*2} \big/ \min\left(c^*, |U|\right)$

- **AUSMPW+**

$\mathbf{F}_{i+1/2,j} = \bar{M}^+_L c_{i+1/2,j}\mathbf{w}_{i,j} + \bar{M}^-_R c_{i+1/2,j}\mathbf{w}_{i+1,j}$ with $\bar{M}^\pm_{L/R} = \bar{M}^\pm_{L/R}(M^+_i, M^-_{i+1}, f_{L/R}, \omega)$

$\to \hat{\mathbf{F}}_{i+1/2,j}$ by $u \to U,\ v \to V$

- **CUSP (H/E-CUSP)**

$\mathbf{D}_{c,i+1/2,j} = \left|u_{i+1/2,j}\right|\Delta\mathbf{w}_{c,i+1/2,j} = f_1\left(M\right)c_{i+1/2,j}\Delta\mathbf{w}_{c,i+1/2,j},\ \ \mathbf{D}_{p,i+1/2,j} = f_2\left(M\right)\Delta p_{i+1/2,j}$

with $M = (M_{i,j} + M_{i+1,j})/2,\ M_{i,j} = u_{i,j}/c_{i,j}\ \ \to\ \hat{\mathbf{F}}_{i+1/2,j}$ by $(u \to U,\ v \to V)$

Chap. 4-5. Explicit Time Integration

- ***Multi-Stage R-K Time-stepping for*** $\dfrac{d\mathbf{U}_{i,j}}{dt} = -\mathbf{R}(\mathbf{U}_{i,j})$
  - **Revisit the multi-stage R-K time-stepping for linear advection equation**
    - See the works by Jameson-Schmidt-Turkel(1981), and others

$$u_t + cu_x = 0 \xrightarrow[\text{to space derivative}]{\text{cenral differencing}} \frac{du_j}{dt} = -\frac{c}{2\Delta x}\left(u_{j+1} - u_{j-1}\right)$$

$$\xrightarrow[\text{ODE in terms of } \omega]{\text{discrete } FT \;\; u_j = \hat{u}(\omega,t)e^{i\omega x_j}} \frac{d\hat{u}(\omega,t)}{dt} = \beta\hat{u}(\omega) \;\text{ with }\; \beta = \text{-}i\frac{c}{\Delta x}\sin\xi,\; \xi = \omega\Delta x \;\rightarrow\; \frac{dy}{dt} = \beta y \;\text{ with }\; \beta \in C$$

- From $y' = \beta y$ with $\beta = \text{-}i\dfrac{a}{\Delta x}\sin\xi$, m-stage explicit R-K scheme is considered.
- To get more stability, add the amount of numerical dissipation such that the locus of β is far away from the positive real axis.
  - Stability for both the negative real axis and the imaginary axis

For $u_t + cu_x = 0$, add 2nd-order numerical diffusion

$$\Delta t\frac{du_j}{dt} = -\frac{\sigma}{2}\left(u_{j+1} - u_{j-1}\right) + \varepsilon^{(2)}\left(u_{j+1} - 2u_j + u_{j-1}\right) \text{ with } \varepsilon^{(2)}\left(u_{j+1} - 2u_j + u_{j-1}\right) \cong \varepsilon^{(2)}\Delta x^2 u_{xx},\; \varepsilon^{(2)} > 0$$

$$\xrightarrow{\text{discrete } FT \;\; u_j = \hat{u}(\omega,t)e^{i\omega x_j}} \frac{d\hat{u}(\omega,t)}{dt} = \frac{1}{\Delta t}\left[-i\sigma\sin\xi - 2\varepsilon^{(2)}\left(1 - \cos\xi\right)\right]\hat{u}(\omega,t) = \beta\hat{u}(\omega,t) \text{ with } \beta = p + iq,\;\; \mathrm{Re}(\beta) = p \le 0$$

or add 4th-order numerical diffusion with $\varepsilon^{(4)}\left(u_{j+2} - 4u_{j+1} + 6u_j - 4u_{j-1} + u_{j-2}\right) \cong \varepsilon^{(4)}\Delta x^4 u_{xxxx}$

$$\xrightarrow{\text{discrete } FT \;\; u_j = \hat{u}(\omega,t)e^{i\omega x_j}} \frac{d\hat{u}(\omega,t)}{dt} = \frac{1}{\Delta t}\left[-i\sigma\sin\xi + 4\varepsilon^{(4)}\left(1 - \cos\xi\right)^2\right]\hat{u}(\omega,t) = \beta\hat{u}(\omega,t) \text{ with } \varepsilon^{(4)} < 0 \text{ to make } \mathrm{Re}(\beta) \le 0$$

Chap. 4-5. Explicit Time Integration

- **Semi-discrete finite volume formulation**

$$\frac{d\mathbf{U}_{i,j}}{dt} = -\mathbf{R}_{i,j} = -\mathbf{R}(\mathbf{U}_{i,j})\left(= -\frac{1}{\Delta S_{i,j}}\sum_{1\sim4}\left(\mathbf{F}\Delta y - \mathbf{G}\Delta x\right)_{1\sim4} \quad \text{or} \quad -\frac{1}{\Delta S_{i,j}}\sum_{1\sim4} T^{-1}\mathbf{F}(\hat{\mathbf{U}})_{1\sim4}\, l_{1\sim4}\right)$$

- **Design of diffusive flux**

$$\mathbf{R}_{i,j} = -\frac{1}{\Delta S_{i.j}}\sum_{1\sim4}\left[T^{-1}\mathbf{F}(\hat{\mathbf{U}})l\right]_{1\sim4} \quad \text{with } \mathbf{F}(\hat{\mathbf{U}})_{i+1/2,j} = \frac{1}{2}\left(\mathbf{F}(\hat{\mathbf{U}})_{i,j} + \mathbf{F}(\hat{\mathbf{U}})_{i+1,j}\right) - \mathbf{D}(\hat{\mathbf{U}})_{i+1/2,j}$$

  - $\mathbf{D}_{i+1/2,j} = \mathbf{D}^{(2)}_{i+1/2,j} - \mathbf{D}^{(4)}_{i+1/2,j}$ is designed to damp out
    - Odd-even decoupling throughout the entire flowfield → weak diffusive flux with *$O(\Delta x^3)$*
    - Oscillations around stiff gradient region → strong diffusive flux with *$O(\Delta x)$*

- $\mathbf{D}^{(2)}_{i+1/2,j} = \varepsilon^{(2)}_{i+1/2,j}\Delta\hat{\mathbf{U}}_{i+1/2,j}$

- $\mathbf{D}^{(4)}_{i+1/2,j} = \varepsilon^{(4)}_{i+1/2,j}\left(\hat{\mathbf{U}}_{i+2,j} - 3\hat{\mathbf{U}}_{i+1,j} + 3\hat{\mathbf{U}}_{i,j} - \hat{\mathbf{U}}_{i-1,j}\right) = \varepsilon^{(4)}_{i+1/2,j}\left(\Delta\hat{\mathbf{U}}_{i+3/2,j} - 2\Delta\hat{\mathbf{U}}_{i+1/2,j} + \Delta\hat{\mathbf{U}}_{i-1/2,j}\right)$

Here, $\varepsilon^{(2)}_{i+1/2,j} = \phi_{i+1/2,j}\min\left(\frac{1}{2}, \nu_{i+1/2,j}\right)$, $\varepsilon^{(4)}_{i+1/2,j} = \phi_{i+1/2,j}\max\left(0, \frac{1}{32} - 2\nu_{i+1/2,j}\right)$

$$\nu_{i+1/2,j} = \max\left(\nu_{i+2,j}, \nu_{i+1,j}, \nu_{i,j}, \nu_{i-1,j}\right),\ \nu_{i,j} = \left|\frac{p_{i+1,j} - 2p_{i,j} + p_{i-1,j}}{p_{i+1,j} + 2p_{i,j} + p_{i-1,j}}\right| \cong \left.\frac{p_{xx}\Delta x^2}{p_{xx}\Delta x^2 + 4p}\right|_{i,j},$$

and $\phi_{i+1/2,j} = \left(|U| + c\right)_{i+1/2,j}$

• From the design of $\varepsilon_{i+1/2,j}$ and $\nu_{i+1/2,j}$, $D_{i+1/2,j}$ is a scalar dissipation based on a pressure sensor.

Chap. 4-5. Explicit Time Integration

- Computatinally very efficient, particularly for transonic steady flows with shocks

  $\rightarrow\ \varepsilon^{(2)}$ and $\varepsilon^{(4)}$ are optimally tuned for transonic flows

- For SCL, a similar form of diffusive flux with

  $d_{i+1/2} = d^{(2)}_{i+1/2} - d^{(4)}_{i+1/2} \cong \varepsilon(\Delta u_{i+1/2} - \alpha(\Delta u_{i+3/2} + \Delta u_{i-1/2}))$ is non-TVD type. Thus, it should be re-formulated by the LED-type limited diffusion as $d_{i+1/2} = \varepsilon(\Delta u_{i+1/2} - L(\Delta u_{i+3/2}, \Delta u_{i-1/2}))$.

- For more accurate (but expensive) computations, scalar diffusion can be replaced by some forms of more elaborate diffusion such as Riemann solvers, AUSP/CUSP-type schemes, and so on.

- **Temporal discretization using a modified 5-stage R-K time integration**

$\mathbf{U}^{n+1}_{i,j} = \mathbf{U}^{n}_{i,j} - \int_{t^n}^{t^n+\Delta t} \mathbf{R}_{i,j}\,dt,$ and $\mathbf{U}^{n+1}_{i,j}$ is updated by evaluating $\int_{t^n}^{t^n+\Delta t} \mathbf{R}_{i,j}\,dt$ as

$$\mathbf{U}^{(0)}_{i,j} = \mathbf{U}^{n}_{i,j},$$

$$\mathbf{U}^{(1)}_{i,j} = \mathbf{U}^{(0)}_{i,j} - \alpha_1 \Delta t_{i,j}\left(\mathbf{E}^{(0)}_{i,j} - \beta_1 \mathbf{D}^{(0)}_{i,j}\right) \text{ with } \mathbf{E}^{(k)}_{i,j} = \frac{1}{2}\left(\mathbf{F}\left(\hat{\mathbf{U}}^{k}_{i,j}\right) + \mathbf{F}\left(\hat{\mathbf{U}}^{k}_{i+1,j}\right)\right) \text{ and } \mathbf{D}^{(k)}_{i,j} = \mathbf{D}\left(\hat{\mathbf{U}}^{k}_{i,j}\right),$$

$$\mathbf{U}^{(2)}_{i,j} = \mathbf{U}^{(0)}_{i,j} - \alpha_2 \Delta t_{i,j}\left(\mathbf{E}^{(1)}_{i,j} - \beta_1 \mathbf{D}^{(0)}_{i,j}\right),$$

$$\mathbf{U}^{(3)}_{i,j} = \mathbf{U}^{(0)}_{i,j} - \alpha_3 \Delta t_{i,j}\left(\mathbf{E}^{(2)}_{i,j} - \left[\beta_3 \mathbf{D}^{(2)}_{i,j} + (1-\beta_3)\mathbf{D}^{(0)}_{i,j}\right]\right),$$

$$\mathbf{U}^{(4)}_{i,j} = \mathbf{U}^{(0)}_{i,j} - \alpha_4 \Delta t_{i,j}\left(\mathbf{E}^{(3)}_{i,j} - \left[\beta_3 \mathbf{D}^{(2)}_{i,j} + (1-\beta_3)\mathbf{D}^{(0)}_{i,j}\right]\right),$$

$$\mathbf{U}^{(5)}_{i,j} = \mathbf{U}^{(0)}_{i,j} - \alpha_5 \Delta t_{i,j}\left(\mathbf{E}^{(4)}_{i,j} - \left[\beta_5 \mathbf{D}^{(4)}_{i,j} + (1-\beta_5)\mathbf{D}^{(2)}_{i,j}\right]\right) \ \rightarrow\ \mathbf{U}^{n+1}_{i,j} = \mathbf{U}^{(5)}_{i,j}.$$

Chap. 4-5. Explicit Time Integration

- From standard R-K framework of linear convection equation, diffusive flux is separately treated, and (α_k , β_k) are determined to provide a maximal stability region along imaginary axis and negative real axis.

  $\alpha_1 = 1/4,\ \alpha_2 = 1/6,\ \alpha_3 = 3/8,\ \alpha_4 = 1/2,\ \alpha_5 = 1$

  $\beta_1 = 1, \beta_3 = 5/9,\ \beta_5 = 4/9\ \left(\beta_2 = \beta_4 = 0\right)$

- **General formulation of a modified *m*-stage R-K scheme**

For $\dfrac{d\mathbf{U}}{dt} = -\mathbf{R}(\mathbf{U})$ with $\mathbf{R}(\mathbf{U}) = \mathbf{E}(\mathbf{U}) - \mathbf{D}(\mathbf{U})$,

$$\mathbf{U}_{i,j}^{(n+1,0)} = \mathbf{U}_{i,j}^{n}$$

…

$$\mathbf{U}_{i,j}^{(n+1,k)} = \mathbf{U}_{i,j}^{(n+1,0)} - \alpha_k \Delta t_{i,j} \left(\mathbf{E}_{i,j}^{(k-1)} - \mathbf{D}_{i,j}^{(k-1)}\right)$$

…

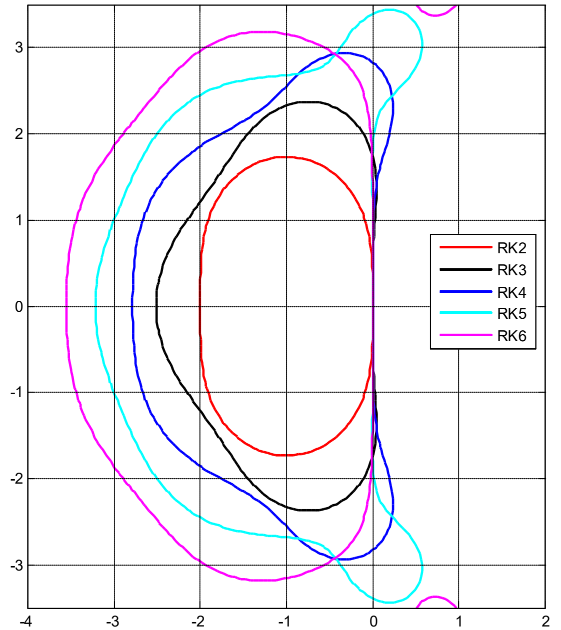
$$\mathbf{U}_{i,j}^{n+1} = \mathbf{U}_{i,j}^{(n+1,m)}$$

Here, k indicates $k-th$ stage and $\alpha_m = 1$

$$\mathbf{E}^{(0)} = \mathbf{E}(\mathbf{U}^n), \mathbf{D}^{(0)} = \mathbf{D}(\mathbf{U}^n)$$

…

$$\mathbf{E}^{(k)} = \mathbf{E}(\mathbf{U}^{(n+1,k)}), \mathbf{D}^{(k)} = \beta_k \mathbf{D}(\mathbf{U}^{(n+1,k)}) + \left(1-\beta_k\right) \mathbf{D}(\mathbf{U}^{(n+1,k-1)})$$



Stability of Standard R-K method

Chap. 4-5. Explicit Time Integration

- From the model equation of the form $u_t + cu_x = c_2 \Delta x^2 u_{xx} + c_4 \Delta x^4 u_{xxxx}$,
  - Apply central differencing to spatial derivatives to model **E(U)** and **D(U)**

$$\rightarrow \Delta x \frac{du_j(t)}{dt} = R(u_{j-2},...,u_j,...u_{j+2})$$

  - Obtain the semi-discrete form in *Fourier* space with $u(x_j,t) = u_j(t) = \hat{u}(\omega,t)e^{i\omega x_j}$

$$\rightarrow \Delta x \frac{d\hat{u}(\omega,t)}{dt} = Z(\xi,c,c_2,c_4)\hat{u}(\omega,t) \text{ with } Z \in C \text{ and } \mathrm{Re}(Z) \le 0$$

  - Apply the modified R-K time integration to determine α_k, β_k such that

    i) α_k is to provide a maximal stability along the imaginary axis.

    ii) β_k is to provide a maximal stability along the negative real axis.

    - Ex) 4-stage modified R-K (4-2 scheme):

      $\alpha_1 = 1/3,\ \alpha_2 = 4/15,\ \alpha_3 = 5/9,\ \alpha_4 = 1$

      $\beta_1 = 1, \beta_2 = 1/2, \beta_3 = \beta_4 = 0$

      5-stage modified R-K (5-3 scheme):

      $\alpha_1 = 1/4,\ \alpha_2 = 1/6,\ \alpha_3 = 3/8,\ \alpha_4 = 1/2,\ \alpha_5 = 1$

      $\beta_1 = 1, \beta_2 = 0, \beta_3 = 5/9, \beta_4 = 0,\ \beta_5 = 4/9$

- The modified R-K schemes with scalar dissipation are particularly effective for steady-state inviscid computations in conjunction with multi-grid method and residual smoothing.
  - Historically, this was the first numerical framework to demonstrate that, i) a converged steady state transonic solution can be obtained by iterative procedure, ii) CFD codes (mostly developed by Jameson) can be successfully applied to practical industrial problems.

Chap. 4-6. Implicit Time Integration

- **From the finite difference/finite volume discretization of the form**

$$\frac{\partial \bar{\mathbf{U}}}{\partial t}+\frac{\partial \bar{\mathbf{F}}}{\partial \xi}+\frac{\partial \bar{\mathbf{G}}}{\partial \eta}=\mathbf{0} \text{ with } \bar{\mathbf{U}}=\frac{\mathbf{U}}{J},\ \bar{\mathbf{F}}=\mathbf{F}y_\eta-\mathbf{G}x_\eta \text{ and } \bar{\mathbf{G}}=-\mathbf{F}y_\xi+\mathbf{G}x_\xi,$$

consider implicit time discretization as $\Delta\bar{\mathbf{U}}^n_{i,j}/\Delta t=-\mathbf{R}(\bar{\mathbf{U}}^n,\bar{\mathbf{U}}^{n+1})$

$$\left(\frac{\bar{\mathbf{U}}^{n+1}_{i,j}-\bar{\mathbf{U}}^{n}_{i,j}}{\Delta t}=\right)\frac{\Delta\bar{\mathbf{U}}^n_{i,j}}{\Delta t}=-\left\{(1-\mu)\left(\frac{\partial \bar{\mathbf{F}}}{\partial \xi}+\frac{\partial \bar{\mathbf{G}}}{\partial \eta}\right)^n+\mu\left(\frac{\partial \bar{\mathbf{F}}}{\partial \xi}+\frac{\partial \bar{\mathbf{G}}}{\partial \eta}\right)^{n+1}\right\}=\mathbf{0} \text{ with } \mu\geq\frac{1}{2}$$

  - $\mu=1$: fully implicit / steady-state calculations
  - $\mu=1/2$: Crank-Nicolson scheme (trapezoidal rule) for 2^{nd}-order accuracy

- **Linearization of the implicit part:** $\bar{\mathbf{F}}^{n+1}=\bar{\mathbf{F}}(\bar{\mathbf{U}}^{n+1}),\ \bar{\mathbf{G}}^{n+1}=\bar{\mathbf{G}}(\bar{\mathbf{U}}^{n+1})$

$$\bar{\mathbf{F}}^{n+1}=\bar{\mathbf{F}}^{n}+\left(\frac{\partial \bar{\mathbf{F}}}{\partial t}\right)\Delta t+O\left(\Delta t^2\right)$$

$$=\bar{\mathbf{F}}^{n}+\left(\frac{\partial \bar{\mathbf{F}}}{\partial \bar{\mathbf{U}}}\right)\left(\frac{\partial \bar{\mathbf{U}}}{\partial t}\right)\Delta t+O\left(\Delta t^2\right)\cong\bar{\mathbf{F}}^{n}+\bar{\mathbf{A}}^n\Delta\bar{\mathbf{U}}^n$$

Similarly, $\bar{\mathbf{G}}^{n+1}\cong\bar{\mathbf{G}}^{n}+\bar{\mathbf{B}}^n\Delta\bar{\mathbf{U}}^n$ with $\bar{\mathbf{A}}^n=\left(\frac{\partial \bar{\mathbf{F}}}{\partial \bar{\mathbf{U}}}\right)^n,\bar{\mathbf{B}}^n=\left(\frac{\partial \bar{\mathbf{G}}}{\partial \bar{\mathbf{U}}}\right)^n$

$$\frac{\Delta\bar{\mathbf{U}}^n_{i,j}}{\Delta t}+\mu\left(\frac{\partial \bar{\mathbf{A}}^n}{\partial \xi}+\frac{\partial \bar{\mathbf{B}}^n}{\partial \eta}\right)\Delta\bar{\mathbf{U}}^n_{i,j}=-\left(\frac{\partial \bar{\mathbf{F}}^n}{\partial \xi}+\frac{\partial \bar{\mathbf{G}}^n}{\partial \eta}\right)_{i,j} \text{ or } \left[\frac{I}{\Delta t}+\mu\left(\frac{\partial \bar{\mathbf{A}}^n}{\partial \xi}+\frac{\partial \bar{\mathbf{B}}^n}{\partial \eta}\right)\right]\Delta\bar{\mathbf{U}}^n_{i,j}=-\mathbf{R}^n \quad \text{Eq.(1)}$$

Chap. 4-6. Implicit Time Integration

- **If $\mu = 1$ and $\Delta t \rightarrow \infty$, we have Newton's iteration for $\frac{\partial \bar{\mathbf{F}}}{\partial \xi} + \frac{\partial \bar{\mathbf{G}}}{\partial \eta} = \mathbf{R}(\bar{\mathbf{U}}) = \mathbf{0}$ to obtain**

$$\mathbf{R}(\bar{\mathbf{U}}^{n+1}) \cong \mathbf{R}(\bar{\mathbf{U}}^{n}) + \frac{\partial \mathbf{R}}{\partial \bar{\mathbf{U}}}\Delta\bar{\mathbf{U}} = \mathbf{0} \;\rightarrow\; \left(\frac{\partial \bar{\mathbf{A}}^n}{\partial \xi} + \frac{\partial \bar{\mathbf{B}}^n}{\partial \eta}\right)\Delta\bar{\mathbf{U}}^n_{i,j} = -\left(\frac{\partial \bar{\mathbf{F}}^n}{\partial \xi} + \frac{\partial \bar{\mathbf{G}}^n}{\partial \eta}\right)_{i,j}$$

- **By splitting the flux Jacobian**

$$\bar{\mathbf{A}}^n = \bar{\mathbf{A}}^+ + \bar{\mathbf{A}}^-, \quad \bar{\mathbf{B}}^n = \bar{\mathbf{B}}^+ + \bar{\mathbf{B}}^- \;\; (\text{ex: } \bar{\mathbf{A}}^\pm = \bar{R}_A \Lambda_A^\pm \bar{R}_A^{-1},\; \bar{\mathbf{B}}^\pm = \bar{R}_B \Lambda_B^\pm \bar{R}_B^{-1})$$

Eq.(1) becomes

$$\left[\frac{\mathbf{I}}{\Delta t} + \mu\left(D_\xi^- \bar{\mathbf{A}}^+ + D_\xi^+ \bar{\mathbf{A}}^- + D_\eta^- \bar{\mathbf{B}}^+ + D_\eta^+ \bar{\mathbf{B}}^-\right)\right]\Delta\bar{\mathbf{U}}^n = -\mathbf{R}^n_{i,j} \qquad \text{Eq.(2)}$$

- **Direct computation of Eq.(2) requires block penta-diagonal matrix inversion.**
  - For 2-D case with N x N mesh and single unknown at each grid point, direct inversion of Eq.(2) $= O(nB^2) = O(N^2N^2) = O(N^4) \rightarrow$ prohibitive computational cost
    - n = No. of unknowns, B = bandwidth