Imperial College London

Development of a balanced adaptive time-stepping strategy based on an implicit JFNK-DG compressible flow solver

Z.-G. Yan, Y. Pan, J. Peiro, S. Sherwin

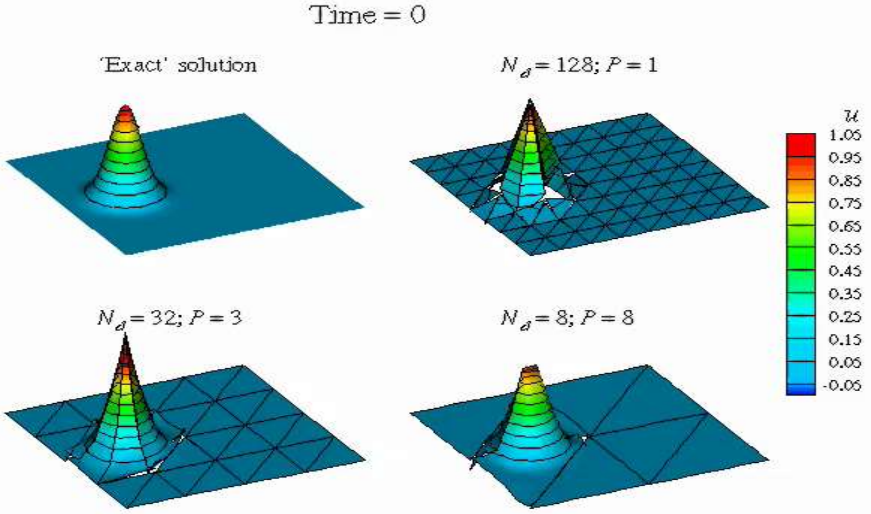
Nekar++ Group

Introduction

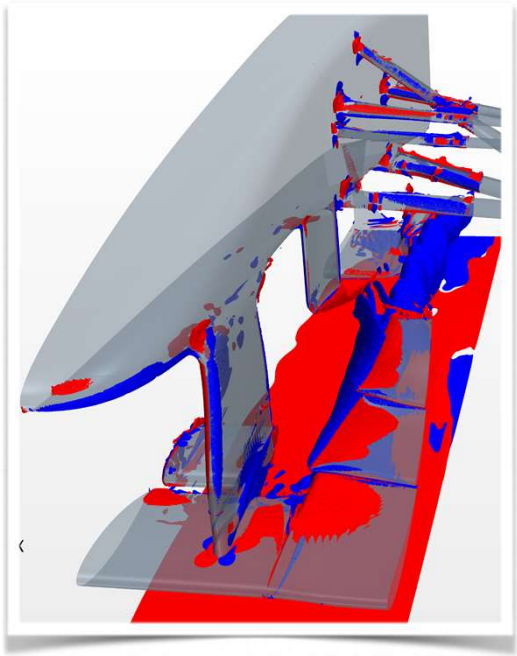
Imperial College London

✓ High-order numerical schemes

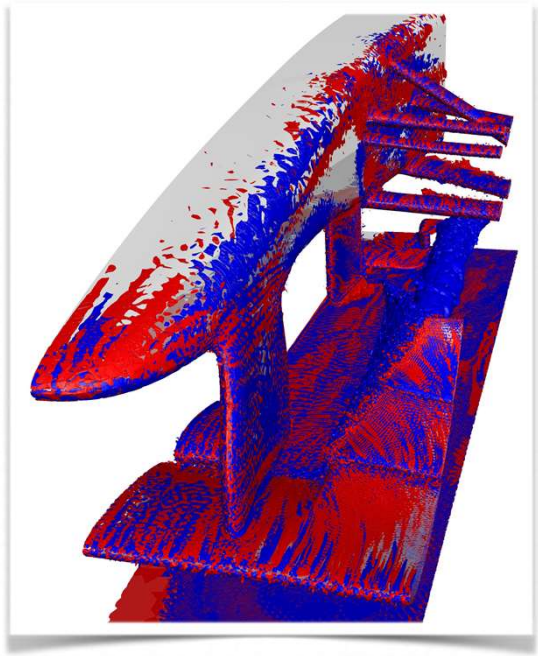
- High convergence rate, high resolution, low dissipation,
- More efficiency in accurate simulations.



Bolis, Cantwell, Kirby, Sherwin Int J Num Meth. fluid, 2014



DES

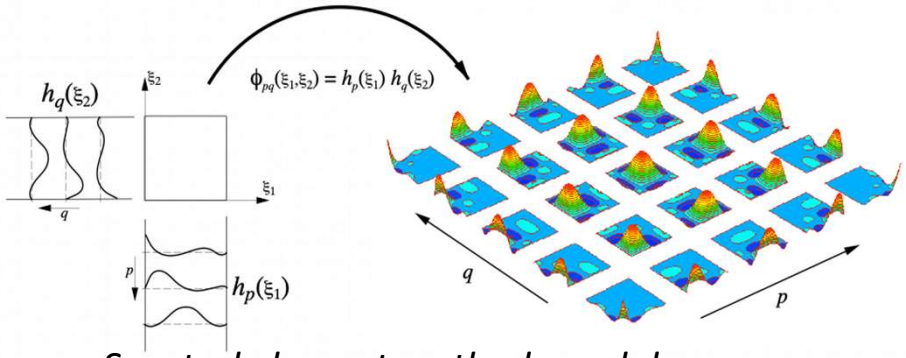


LES

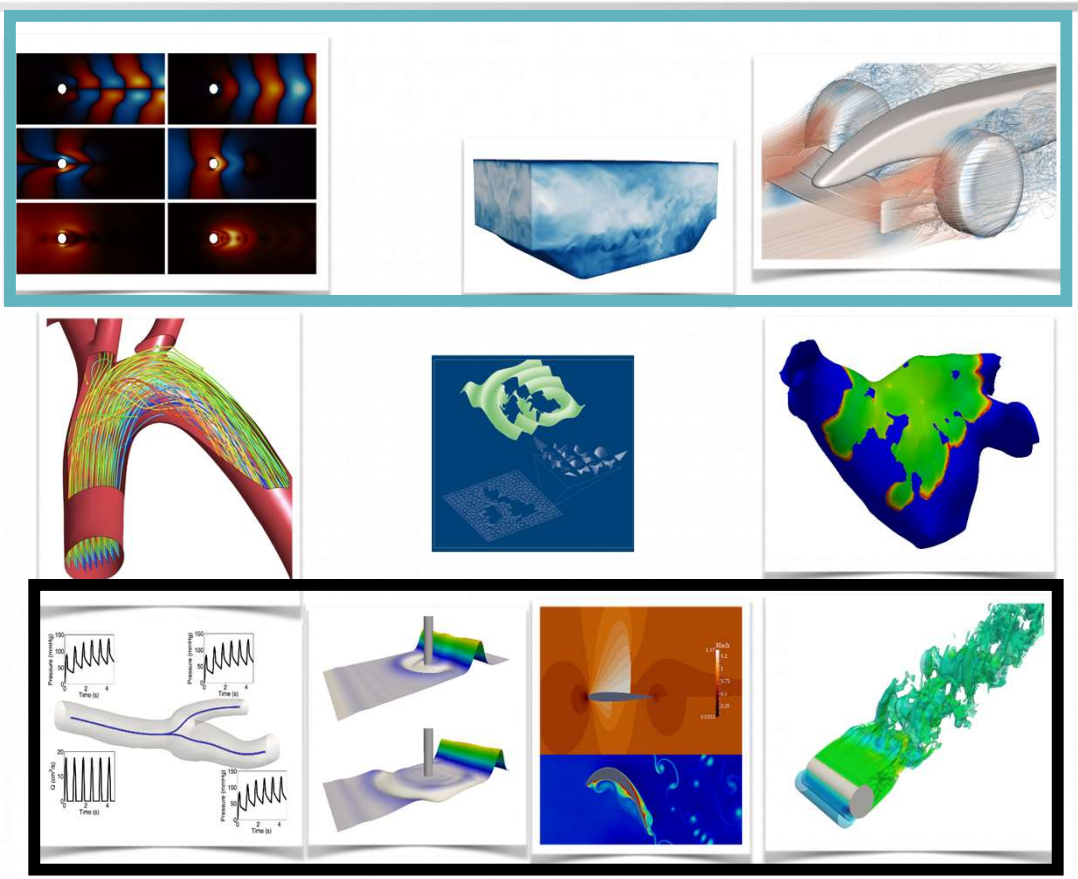
Introduction

Imperial College London

✓ Spectral/hp element methods



Spectral element method - nodal



Nektar++: www.nektar.info

Introduction

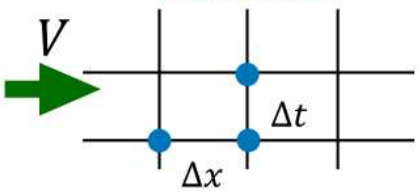
✓ Why Implicit?

- More expensive than FD,
- More severe time step restriction,
- Implicit time stepping: relax the restriction,

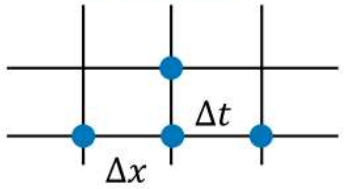
Explicit Δt restrictions:

$$\frac{\partial u}{\partial t} = -V\frac{\partial u}{\partial x} + \sigma\frac{\partial^2 u}{\partial x^2}$$

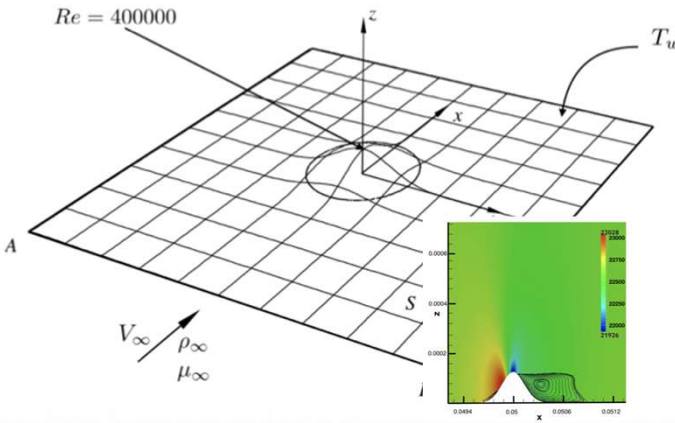
Advection



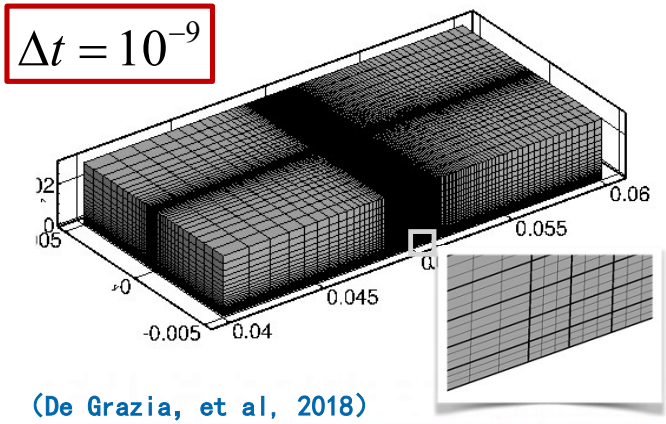
Diffusion



Finite Difference	Spectral/hp Element
$\Delta t \leq C\frac{\Delta x}{V}$	$\Delta t \leq C\frac{\Delta x}{Vp^2}$
$\Delta x \Rightarrow \frac{\Delta x}{p^2}$	
$\Delta t \leq C\frac{\Delta x^2}{\sigma}$	$\Delta t \leq C\frac{\Delta x^2}{\sigma p^4}$



$\Delta t = 10^{-9}$



(De Grazia, et al, 2018)

Implicit JFNK-DG solver

Imperial College London

✓ Governing equations

- NS equations

$$\frac{\partial \mathbf{U}}{\partial t} + \sum_{i=1}^{d} \frac{\partial \mathbf{H}_i}{\partial \mathbf{x}_i} = 0,$$

$$\mathbf{U} = \sum_{p=0}^{N_P} \phi_p(\mathbf{x}) \mathbf{u}_p(t)$$

✓ Spatial discretization

- Multiply ϕ and integrate by parts:

$$\frac{d\mathbf{u}}{dt} = L\mathbf{u}$$

$$= \mathbf{M}^{-1}\left(\int_{\Omega} \nabla\phi \cdot \mathbf{H} d\Omega - \int_{\Gamma} \phi \mathbf{H}^n d\Gamma\right),$$

- Spatial truncation error

$$\mathbf{e}_s(\mathbf{u}_\delta) = L_\delta \mathbf{u}_\delta - L\mathbf{u}$$

$$= C_s D_s(\mathbf{u}) h^{P+1} + O(h^{P+2}),$$

Implicit JFNK-DG solver

Imperial College London

✓ Diagonally implicit Runge-Kutta

- Multi-stage Runge-Kutta

$$\mathbf{s}^{(i)} = \mathbf{u}^n + \Delta t \sum_{j=1}^{i-1} a_{ij} L\mathbf{u}^{(j)},$$

$$\mathbf{u}^{(i)} = \mathbf{s}^{(i)} + \Delta t a_{ii} L\mathbf{u}^{(i)},$$

nonlinear system

$$\mathbf{u}^{n+1} = \mathbf{u}^n + \Delta t \sum_{i=1}^{s} b_i L\mathbf{u}^{(i)},$$

$$\hat{\mathbf{u}}^{n+1} = \mathbf{u}^n + \Delta t \sum_{i=1}^{s+1} \hat{b}_i L\mathbf{u}^{(i)},$$

- Embedded scheme for error estimate

$$\begin{aligned}\mathbf{e}_t^{n+1} &= \mathbf{u}^{n+1} - \hat{\mathbf{u}}^{n+1} \\ &= \Delta t \sum_{i=1}^{s+1} (b_i - \hat{b}_i) L\mathbf{u}^{(i)} \\ &= C_t D_t(\mathbf{u}) \Delta t^{N+1} + O(\Delta t^{N+2}),\end{aligned}$$

Implicit JFNK-DG solver

Imperial College London

✓ Jacobian-free Newton Krylov

- Nonlinear system for implicit stages

$$\mathbf{N}(\mathbf{u}_\delta^{(i)}) = \mathbf{u}_\delta^{(i)} - \mathbf{s}^{(i)} - \Delta t a_{ii} L_\delta \mathbf{u}_\delta^{(i)} = \mathbf{0}.$$

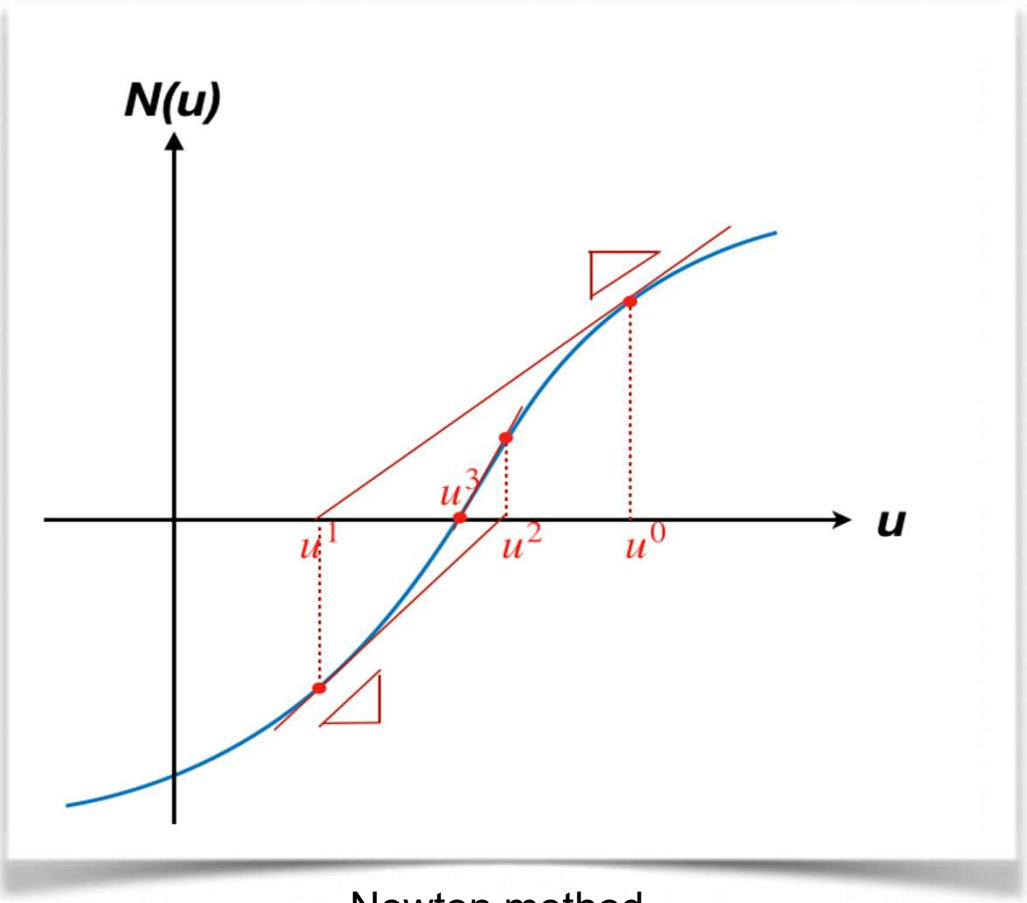
- Newton method

$$\frac{\partial \mathbf{N}(\mathbf{v}^k)}{\partial \mathbf{v}} \Delta \mathbf{v} = -\mathbf{N}(\mathbf{v}^k).$$

- Newton convergence criteria：

$$\| \mathbf{N}(\mathbf{v}^k) \| = \| \mathbf{R}(\mathbf{v}^k) \| \le \tau,$$

Iteration Error



Newton method

Implicit JFNK-DG solver

✓ Jacobian-free Newton Krylov

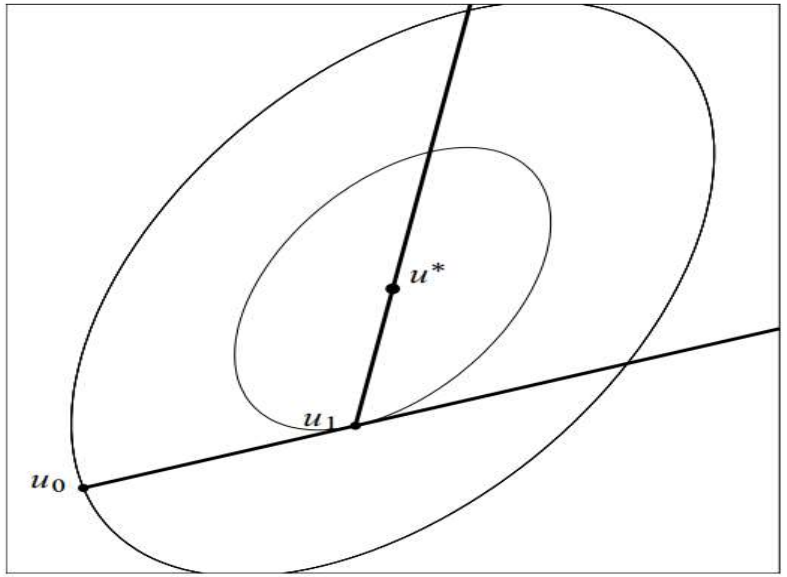
- Linear solver：GMRES

1. calculate q_n with the Arnoldi method;
2. find the y_n which minimizes $\|r_n\|$;
3. compute $x_n = Q_n y_n$;
4. repeat if the residual is not yet small enough.

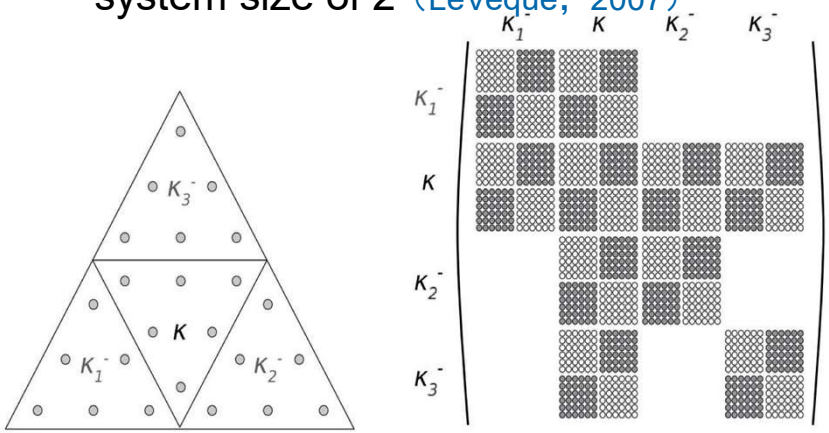
- Jacobian-free matrix vector inner product

$$\frac{\partial \mathbf{N}(\mathbf{v}^k)}{\partial \mathbf{v}}\mathbf{q}_i = \frac{\mathbf{N}(\mathbf{v}^k + \xi \mathbf{q}_i) - \mathbf{N}(\mathbf{v}^k)}{\xi},$$

- Easy coding, low error and low storage.



Diagrammatic diagram of Krylov with system size of 2（LeVeque，2007）



Grid and Jacobian structure

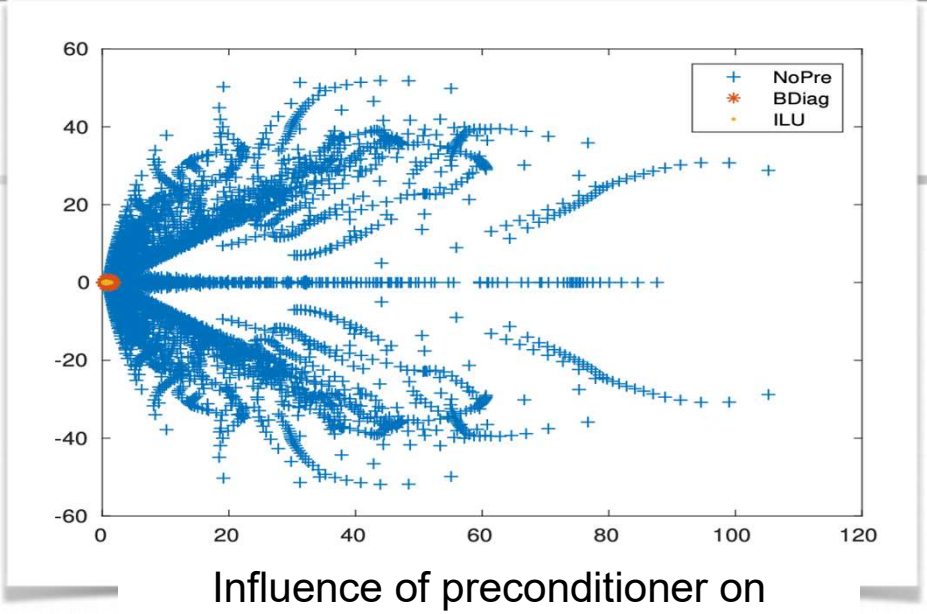
Implicit JFNK-DG solver

✓ Jacobian-free Newton Krylov

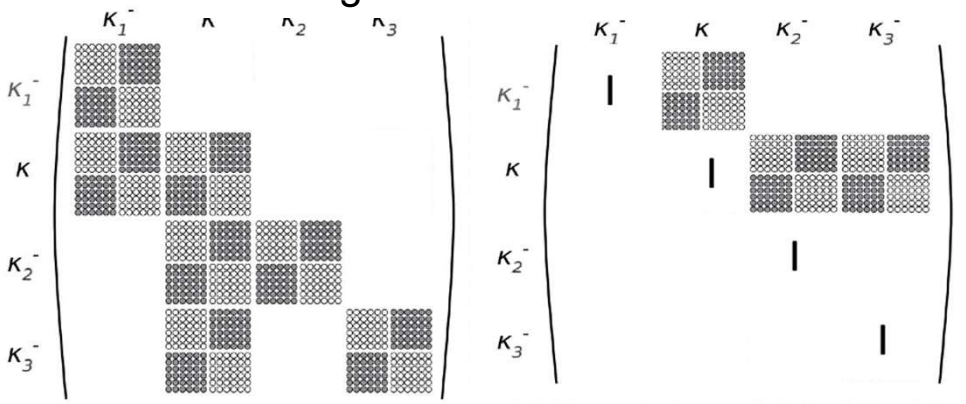
- Preconditioning

$$\left(\frac{\partial \mathbf{N}(\mathbf{v}^k)}{\partial \mathbf{v}}\mathbf{P}^{-1}\right)(\mathbf{P}\Delta\mathbf{v})\,\Delta\mathbf{v} = -\mathbf{N}(\mathbf{v}^k).$$

- Preconditioning matrix: usually matrix based, freezing to lower cost.
- Cost and storage increase quickly $p^{6\text{-}8}$。



Influence of preconditioner on eigenvalue distribution



ILU preconditioner

Implicit JFNK-DG solver

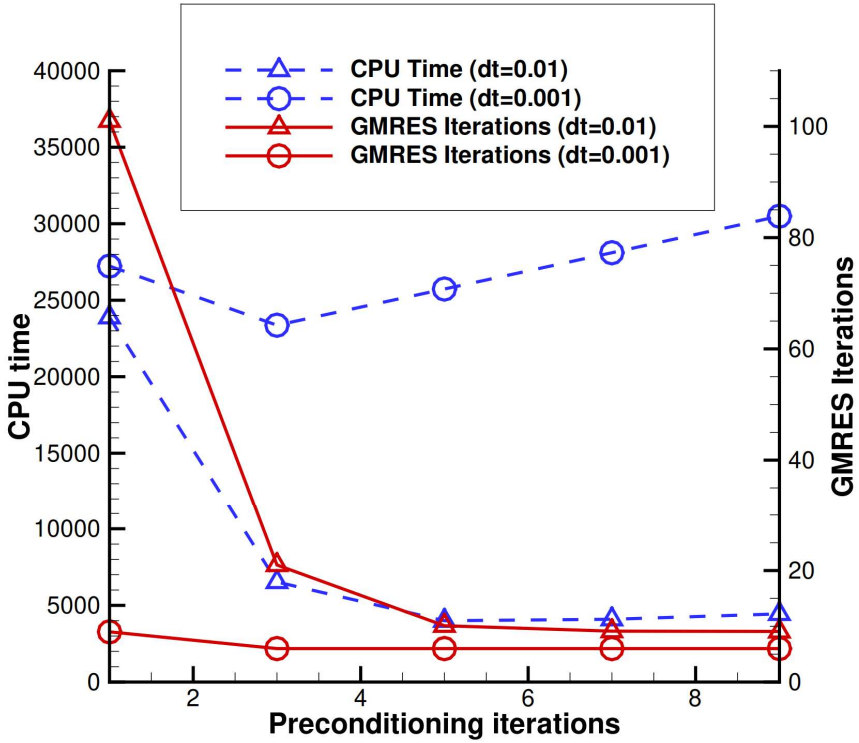
Imperial College London

✓ Preconditioner

- Block relaxed (weighted) Jacobi (BRJ)
  - $\hat{\mathbf{q}}^{j} = \hat{\mathbf{D}}^{-1}\left[\mathbf{q} - \left(\hat{\mathbf{L}} + \hat{\mathbf{U}}\right)\hat{\mathbf{q}}^{j-1}\right]$,
  - Partly matrix-free,
  - 10 times lowers in storage than ILU,
  - 5 times more efficient than diagonal preconditioner (similar storage).

Table 2
Memory consumption of BRJ in bytes for each hexahedral mesh.

	BRJ-diag	BRJ-offdiag	BRJ-total	Jacobian-dense	Jacobian-sparse
$P = 2$	72 900	30 000	102 900	1 020 600	712 800
$P = 3$	409 600	43 200	452 800	5 734 400	3 328 000
$P = 4$	1 562 500	76 800	1 639 300	21 875 000	11 000 000



Implicit JFNK-DG solver

Imperial College London

✓ Summary

Discretise

Nonlinear system

Compressible flow equations

$$\frac{\partial \mathbf{U}}{\partial t} + \nabla \cdot \mathbf{F} = \nabla \cdot \mathbf{G}$$

DG Spatial discretisation

$$\mathbf{M}\frac{d\mathbf{U}}{dt} = -\nabla \cdot \mathbf{F}^{\delta} + \nabla \cdot \mathbf{G}^{\delta} = \mathscr{L}(\mathbf{U}, \nabla \mathbf{U})$$

Implicit time discretisation

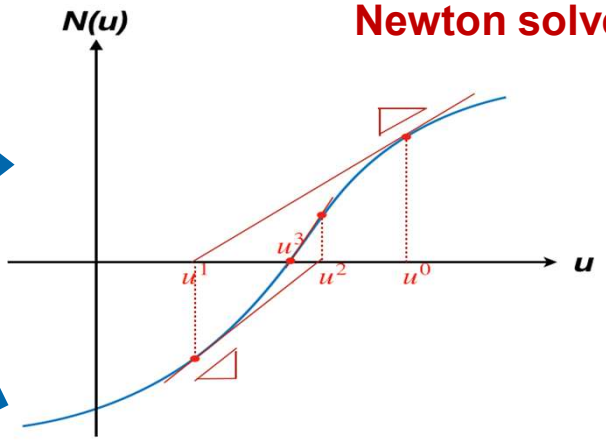
$$\mathbf{U}^{n+1} = \mathbf{U}^{n} + \Delta t \mathbf{M}^{-1} \mathscr{L}(\mathbf{U}^{n+1}, \nabla \mathbf{U}^{n+1})$$

Non-linear system

$$\mathbf{N}(\mathbf{U}^{n+1}) = \mathbf{U}^{n+1} - \mathbf{U}^{n} - \Delta t \mathbf{M}^{-1} \mathscr{L}(\mathbf{U}^{n+1}, \nabla \mathbf{U}^{n+1}) = \mathbf{0}$$

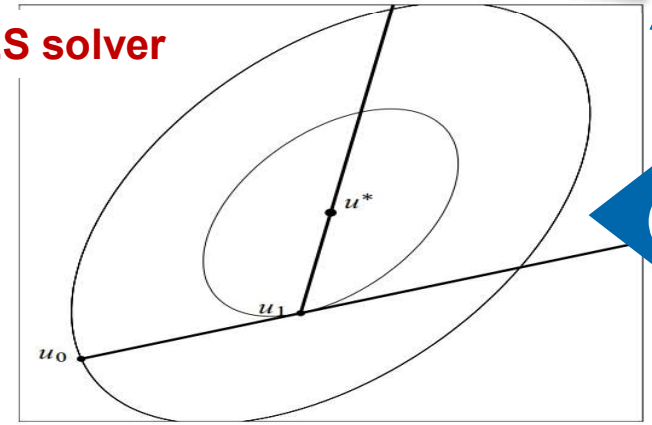
N(u)=0

Newton solver

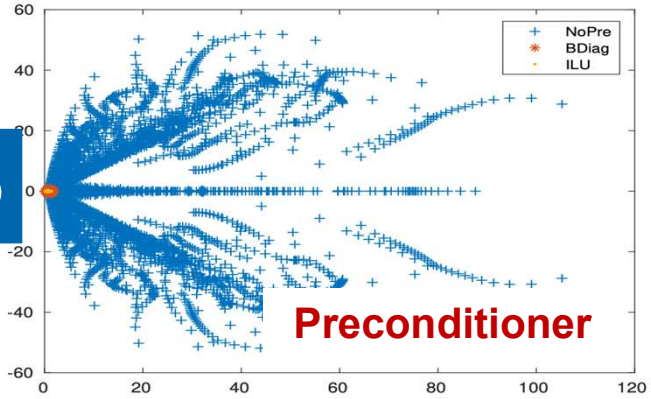


Ax=b

GMRES solver



$(AP^{-1})(Px)=b$



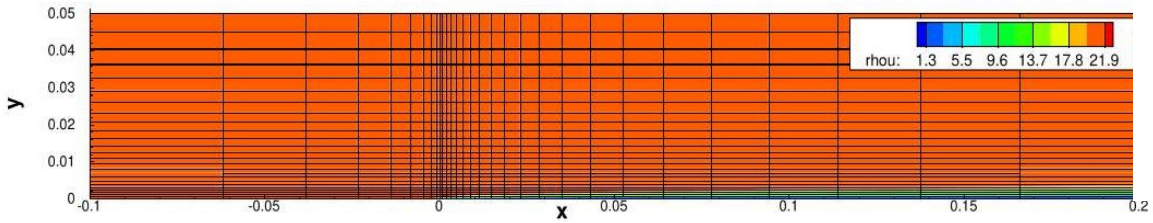
Preconditioner

Implicit JFNK-DG solver

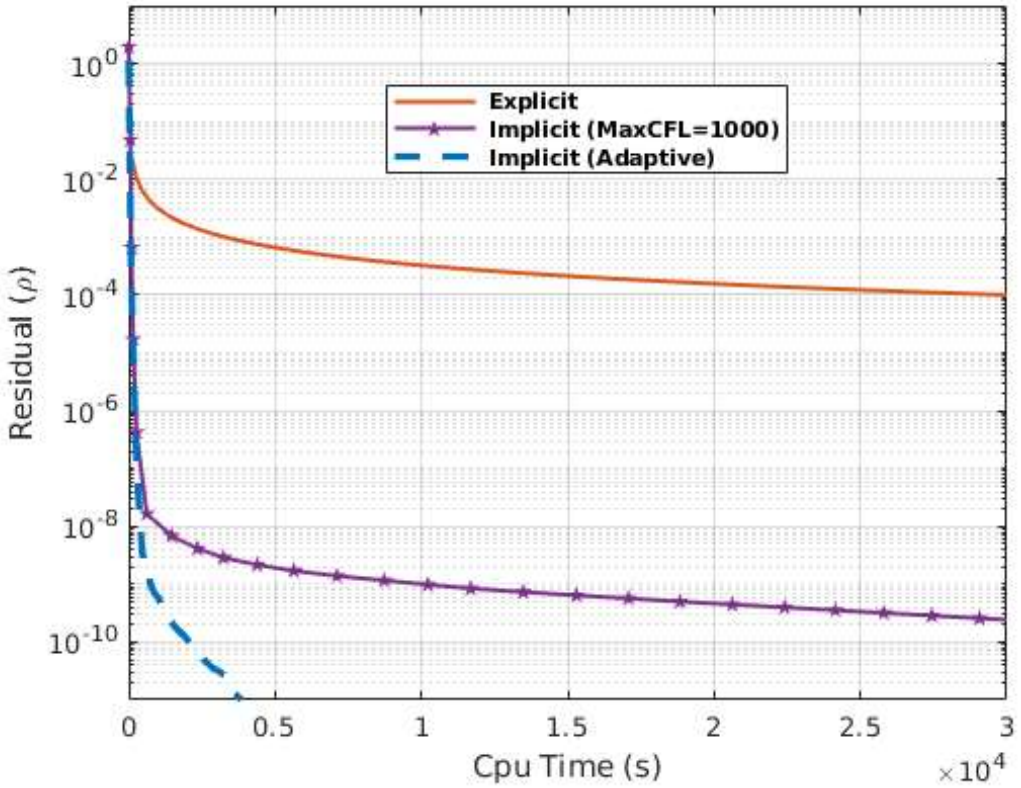
Imperial College London

✓ Implicit solver in steady problem

- Steady flat plate,
- Ma=0.1, Re=10^6,
- More than 3 orders of magnitude higher in efficiency.



Grid and result



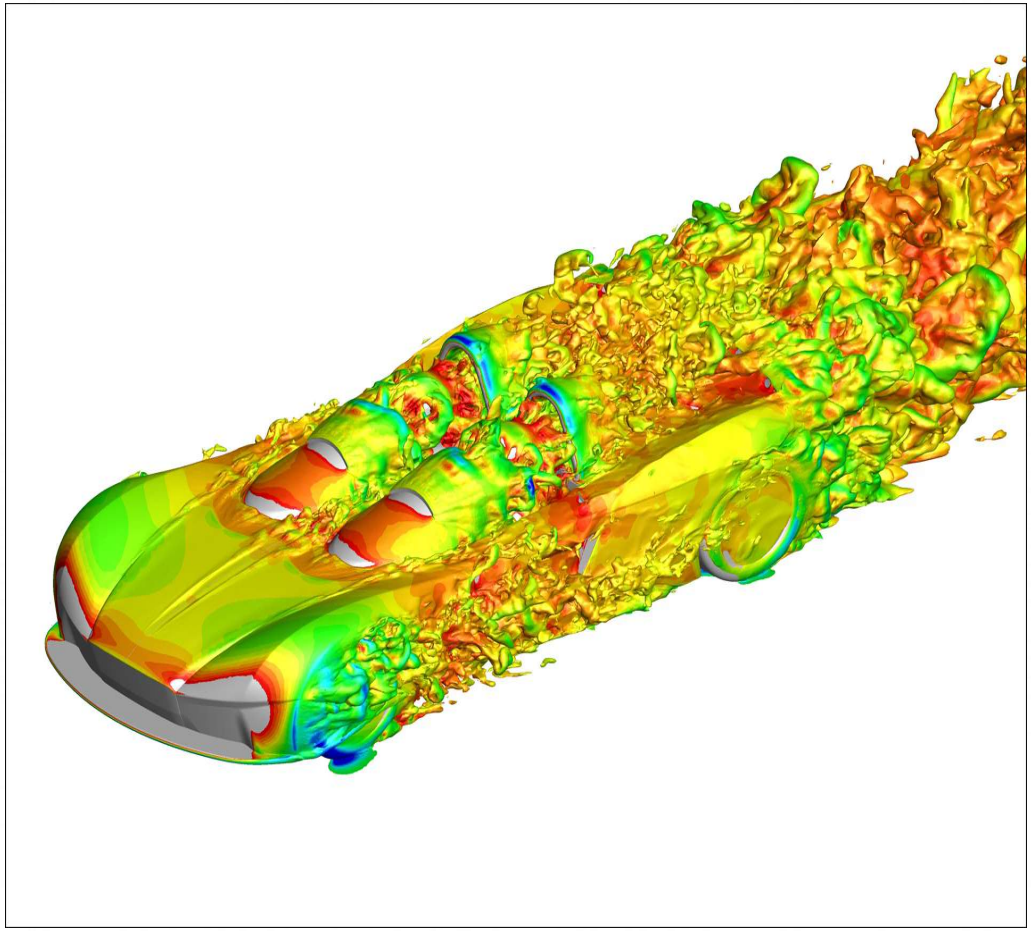
Convergence history

Balanced time step adaption

Imperial College London

✓ Difference in unsteady simulations

- Temporally accurate:
  - ➢ Time step,
  - ➢ Temporal scheme and order,
  - ➢ Convergence of Newton iteration.
- Matrix update.

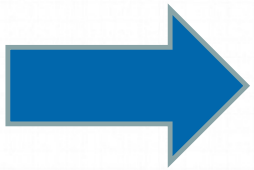


Balanced time step adaption

Imperial College London

✓ Questions of implicit method

- How to choose the time step?
- Is it accurate with such a large time step?
- How is the tolerance determined?
- Is the parameters problem dependent?
- Do we need high-order temporal schemes?
- ….



Numerical experiments

- Not user friendly,
- Problem dependent.

Balanced time step adaption

Imperial College London

✓ When accurate enough in time?

Scheme	Exp 3rd-order	Imp 3rd-order
Time step	2.4E-5	2.7E-2
Temporal error	e_t	~$10^{3X3}e_t$
CPU time	1.0E5	2.5E3
Strouhal	0.2438	0.2438

- When accurate enough?
  - No obviously influence on target value,
  - Condition for implicit to be more efficient,
  - Problem dependent, no known before hand,

- Condition in discrete system?
  - When temporal errors has no obviously influence the discrete solution,
  - Temporal error << spatial error.

Balanced time step adaption

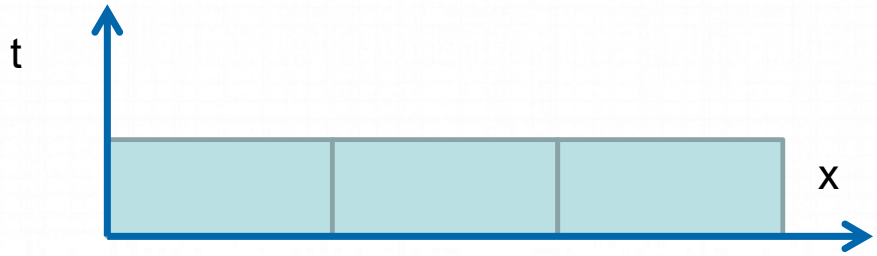
Imperial College London

✓ Balanced temporal error

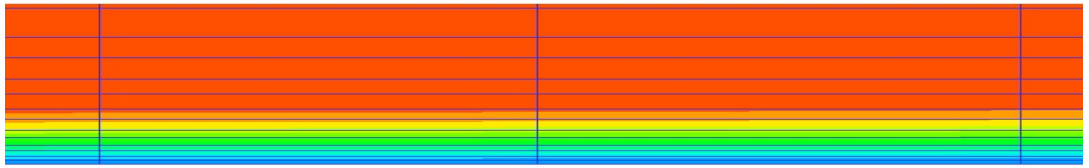
- Large influence on efficiency
  - ➢ Up to 96% cost waste if not balanced

    （G. Noventa,et al, 2016）
  - ➢ Similar to the balancing of spatial errors.

	Spatial order	CPU time [wu]	$\|err_p\|_2$
ESDIRK	$\mathbb{P}^6$	7.10E+02	1.35E−02
	$\mathbb{P}^5$	3.65E+02	1.35E−02
	$\mathbb{P}^4$	1.85E+02	1.35E−02
	$\mathbb{P}^3$	8.25E+01	1.35E−02
	$\mathbb{P}^2$	2.85E+01	1.35E−02
	$\mathbb{P}^1$	1.56E+01	1.32E−02
	$\mathbb{P}^0$	1.89E+01	1.87E−01



Balancing spatial and temporal error



Balancing errors from different directions in BL

Balanced time step adaption

Imperial College London

✓ Adaptive time stepping

- Local error estimate

$$\mathbf{u}_{it}^{n+1} - \mathbf{u}(t^{n+1})$$

$$\approx \Delta t \sum_{i=1}^{s+1} (b_i - \hat{b}_i) L\mathbf{u}^{(i)} + \Delta t \sum_{i=1}^{s} b_i (L_\delta \mathbf{u}_\delta^{(i)} - L\mathbf{u}^{(i)})$$

$$+ \Delta t \sum_{i=1}^{s} b_i (L_\delta \mathbf{u}_{it}^{(i)} - L_\delta \mathbf{u}_\delta^{(i)})$$

$$\approx C_t D_t(\mathbf{u})(\Delta t)^{N+1} + \Delta t C_s \overline{D_s}(\mathbf{u}) h^{P+1} + \Delta t \bar{\mathbf{e}}_{it}$$

$$\approx \mathbf{e}_t^{n+1} + \Delta t \bar{\mathbf{e}}_s + \Delta t \bar{\mathbf{e}}_{it},$$

Should be decreased simultaneously !!

- Spatial error estimate

$$\bar{\mathbf{e}}_s \approx L_\delta^{P+1} \mathbf{u}_{it}^{n+1} - L_\delta^P \mathbf{u}_{it}^{n+1}$$

$$= C_s D_s(\mathbf{u}) h^{P+1} + O(h^{P+2}),$$

- Temporal error estimate

$$\mathbf{e}_t^{n+1} = \mathbf{u}^{n+1} - \hat{\mathbf{u}}^{n+1}$$

$$= C_t D_t(\mathbf{u}) \Delta t^{N+1} + O(\Delta t^{N+2}),$$

Balanced time step adaption

Imperial College London

✓ Adaptive time stepping

- Condition for temporally accurate

$$\| \mathbf{e}_t^{n+1} \| = \beta \Delta t \| \bar{\mathbf{e}}_s \|$$

Temporal error not dominating

- Elemental time step

$$\Delta t_{e,m}^{n+1} = \Delta t^n \left(\frac{\beta \Delta t \| \bar{\mathbf{e}}_s \|_{e,m} + 1.5^N \epsilon_m}{\| \mathbf{e}_t \|_{e,m} + \epsilon_m} \right)^{1/N},$$

- Overall time step

$$\Delta t_m^{n+1} = \sum_{e=1}^{N_e} \| \mathbf{e}_t \|_{e,m}^r \Delta t_{e,m}^{n+1} / \sum_{e=1}^{N_e} \| \mathbf{e}_t \|_{e,m}^r.$$

$$\Delta t^{n+1} = \min(\Delta t_m^{n+1})$$

- Newton convergence criteria

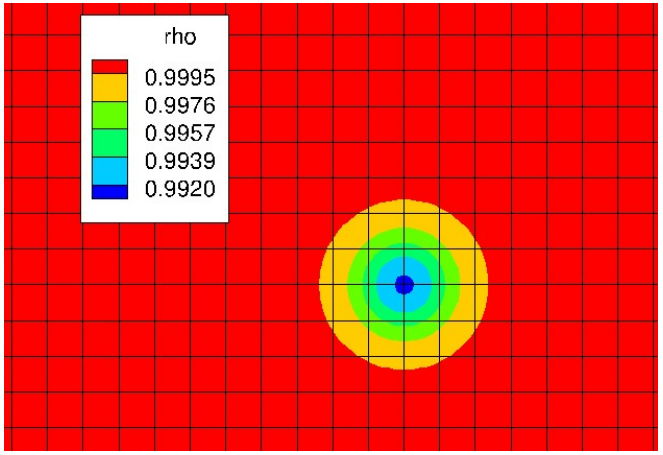
$$\tau = \eta \| \mathbf{e}_t \|,$$

Iterative error not dominating

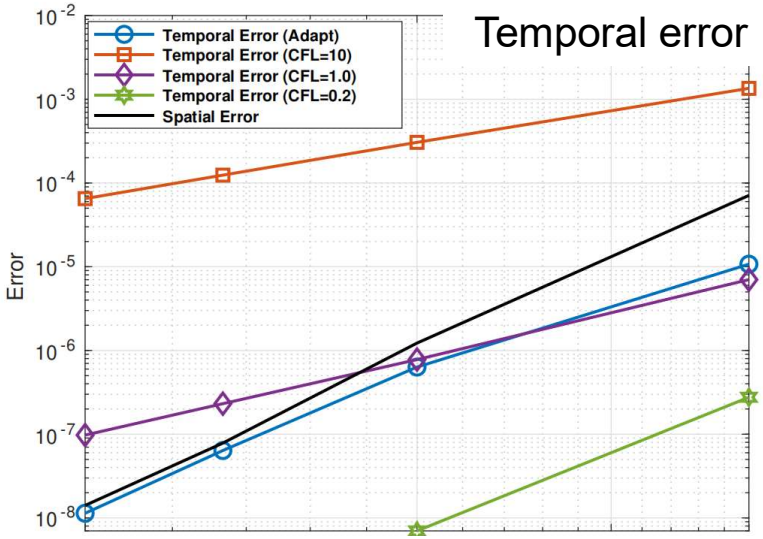
Numerical tests

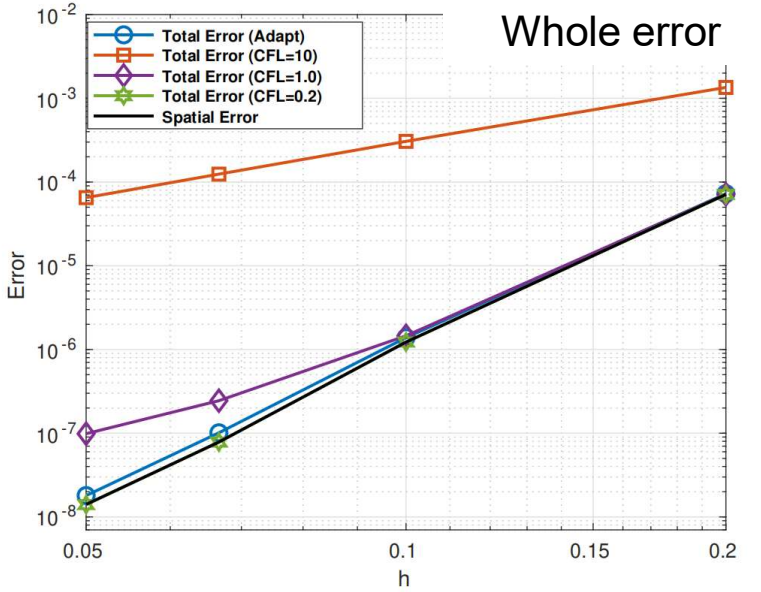
✓ Isentropic vortex

- Temporal error is smaller,
- Temporal error close to the maximum value without obviously influencing the whole error.



Mesh and density



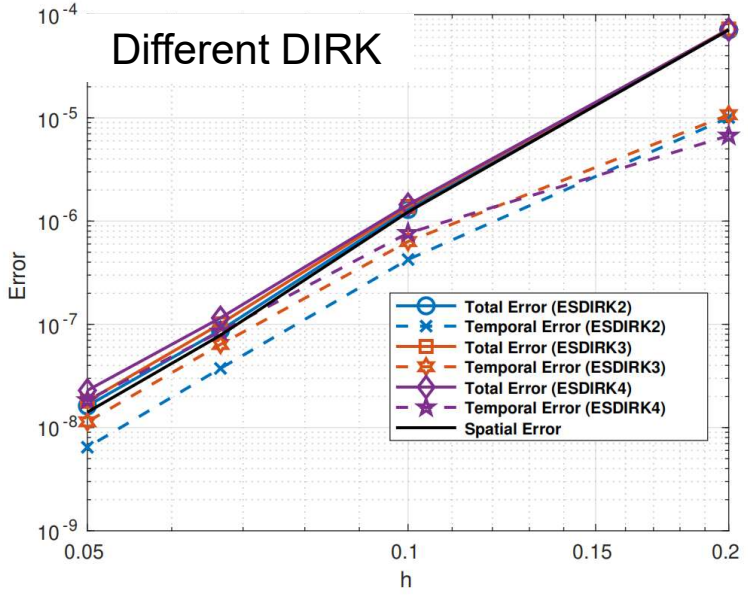


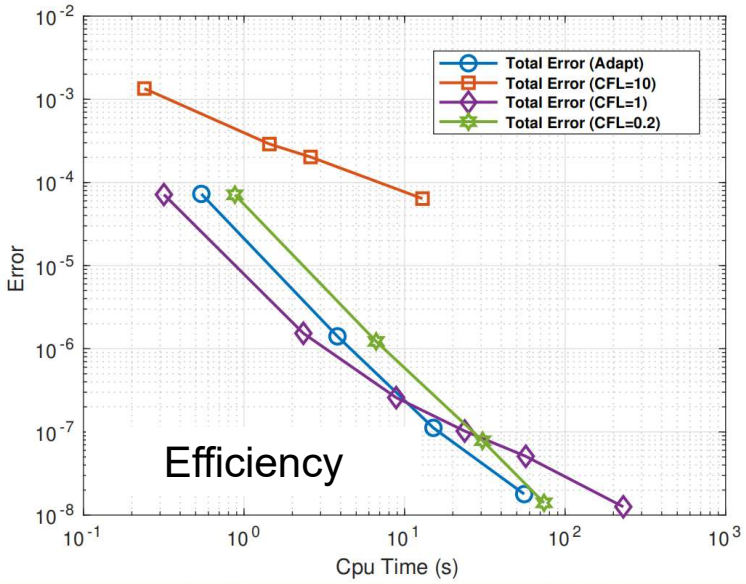
Numerical tests

Imperial College London

✓ Isentropic vortex

- Independent of temporal scheme,
- Close to most efficient choice at all error levels,
- Up to 10 times more efficient than a naive choice.

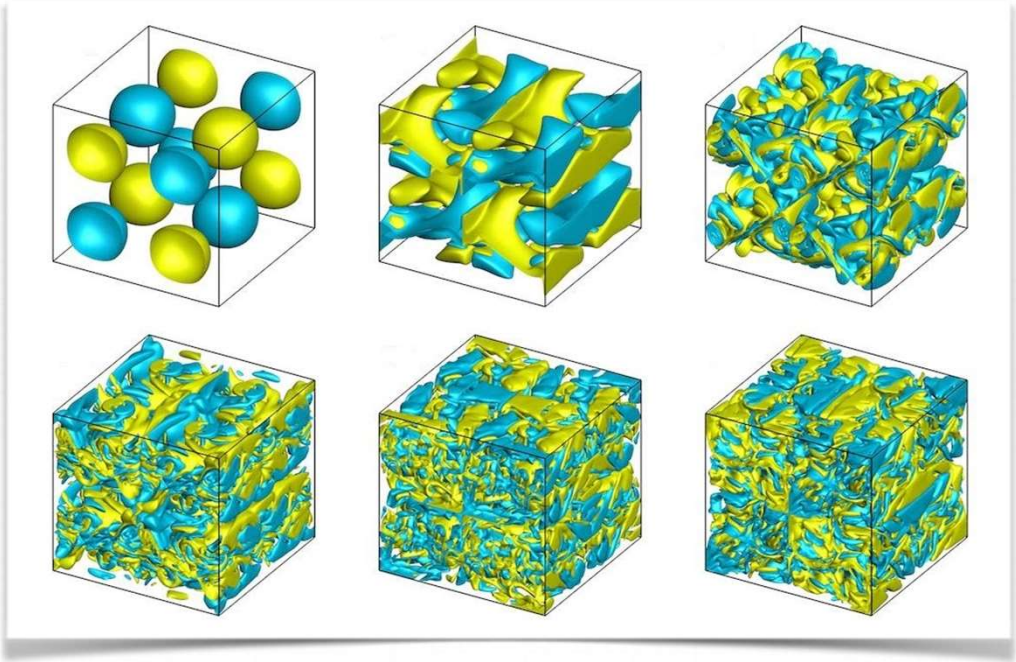


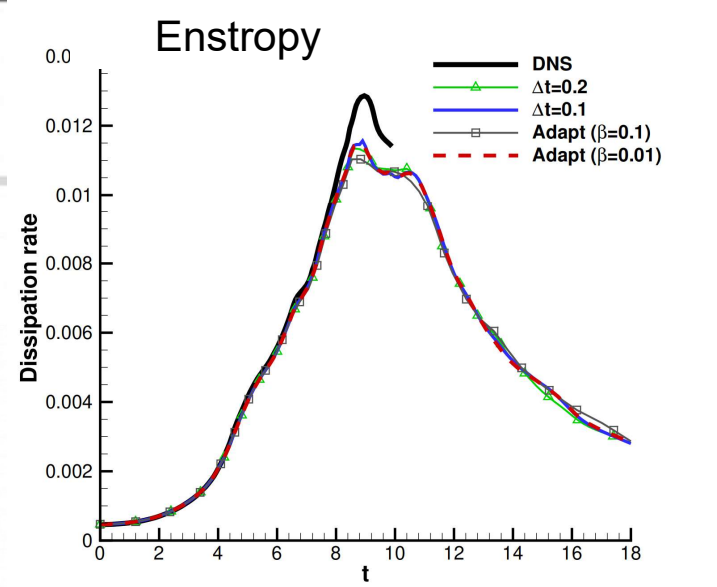


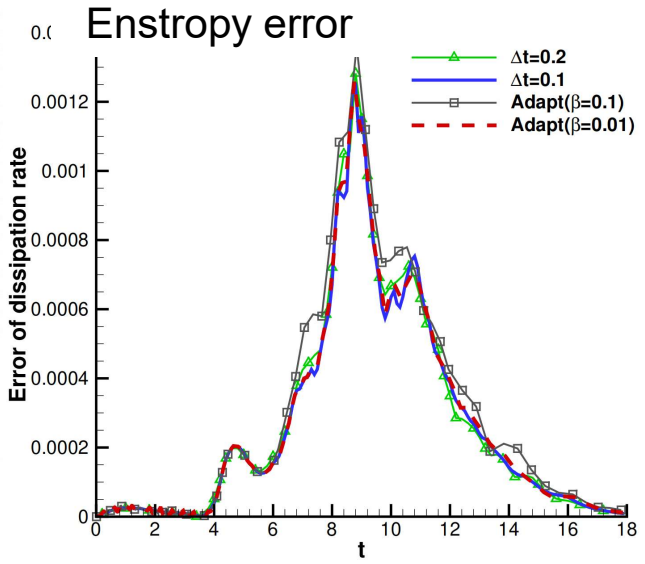
Numerical tests

✓ Taylor-Green vortex

- Temporal accuracy maintained,
- Close to optimal in efficiency.



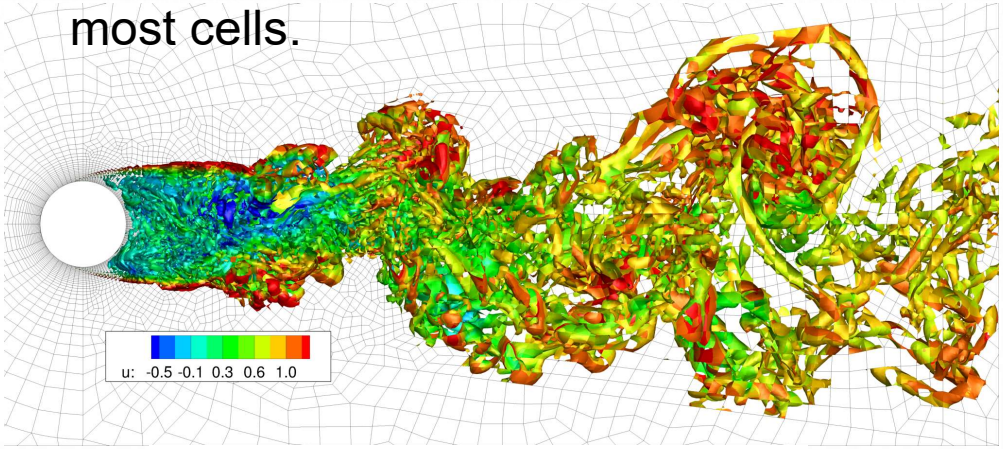




Numerical tests

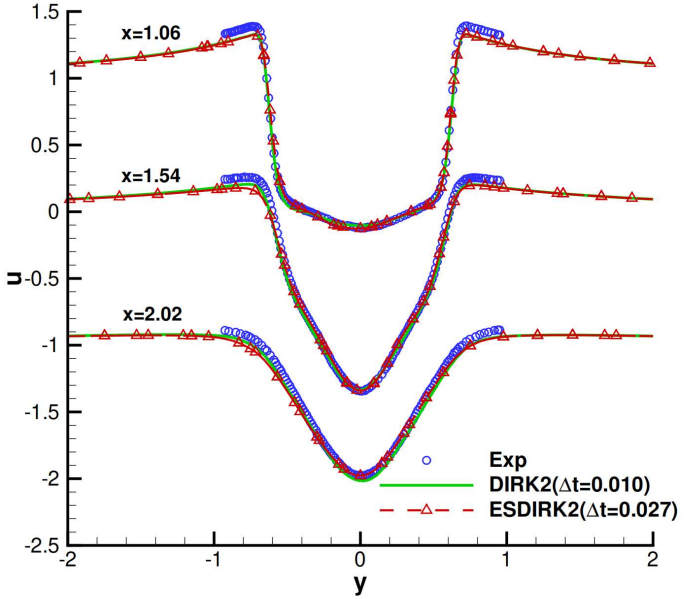
✓ Flow over cylinder

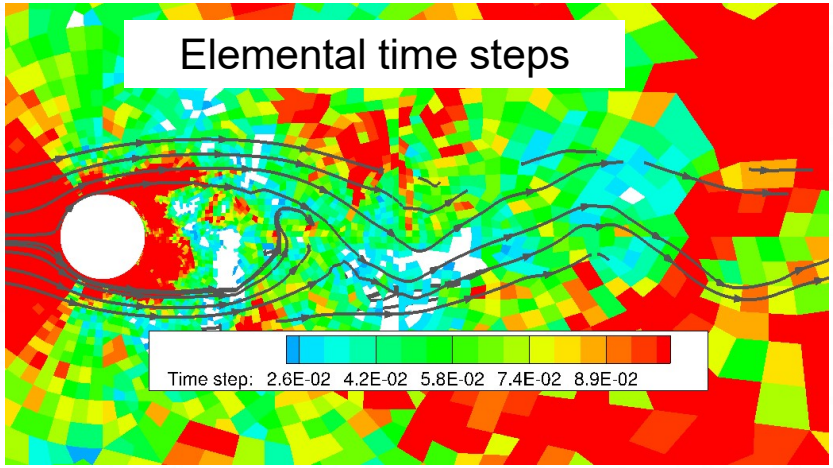
- Temporal accuracy maintained,
- Close to optimal efficiency,
- 14 times speedup compared to explicit method,
- Spatial and temporal error relation maintained in most cells.



Mesh and vortex structure

Time averaged velocity

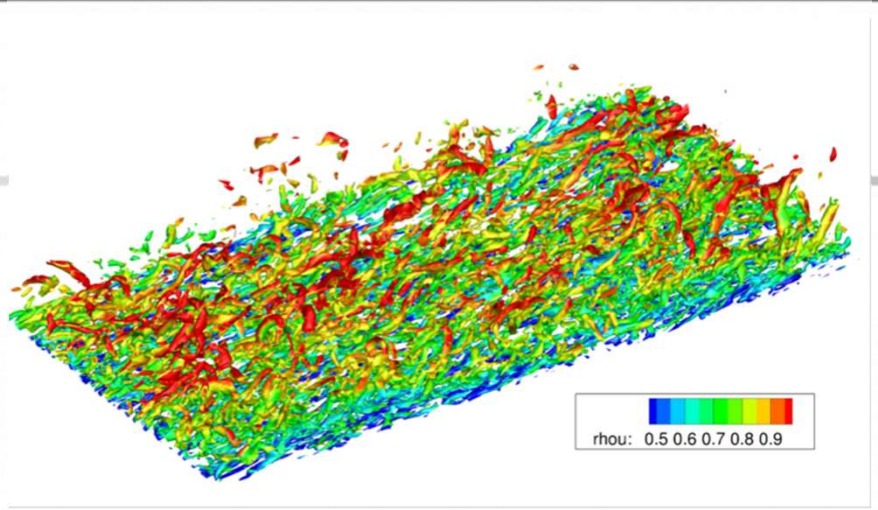


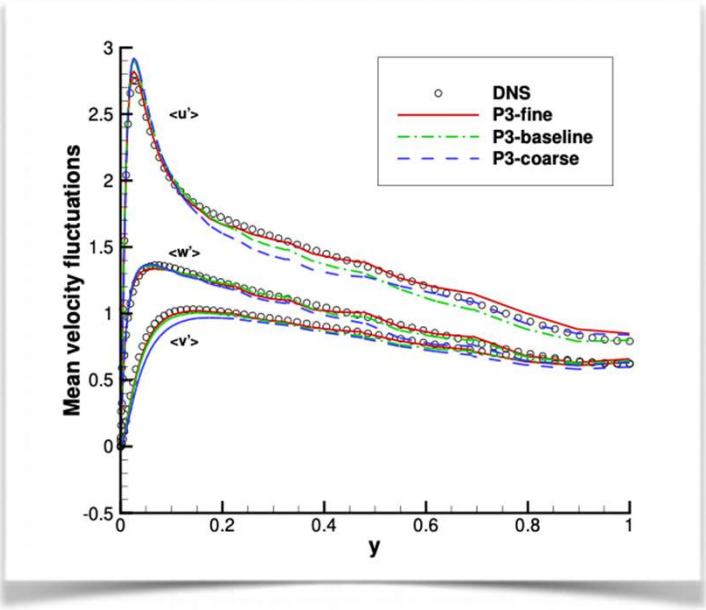


Numerical tests

✓ Turbulent channel flow

	RK2	DIRK2
Δt	4.6×10^{-5}	1.2×10^{-2}
CFL	0.15	40
CFLc	0.0019	0.5
Speed up	1.0	10.3





Concluding remarks

Imperial College London

- Adaptive time stepping balancing spatial and temporal errors,
- Efficient since avoiding too small time steps,
- Applicable to a wide range of problems,
- Save the cost in obtaining a case by case time step,
- User friendly,
- Important for software of wide user and application area like Nektar++.

Thanks! & Questions?

Nekar++ Group