Bifurcation Analysis for Time Steppers

Laurette Tuckerman

laurette@pmmh.espci.fr

Bergeon, Boronska, Brynjell-Rahkola, Feudel, Gonzalez Sembla, Mamun, Rambert, Riquier, …

THE THREE TOOLS OF ~~COMPUTATIONAL~~ Geophysical FLUID DYNAMICS

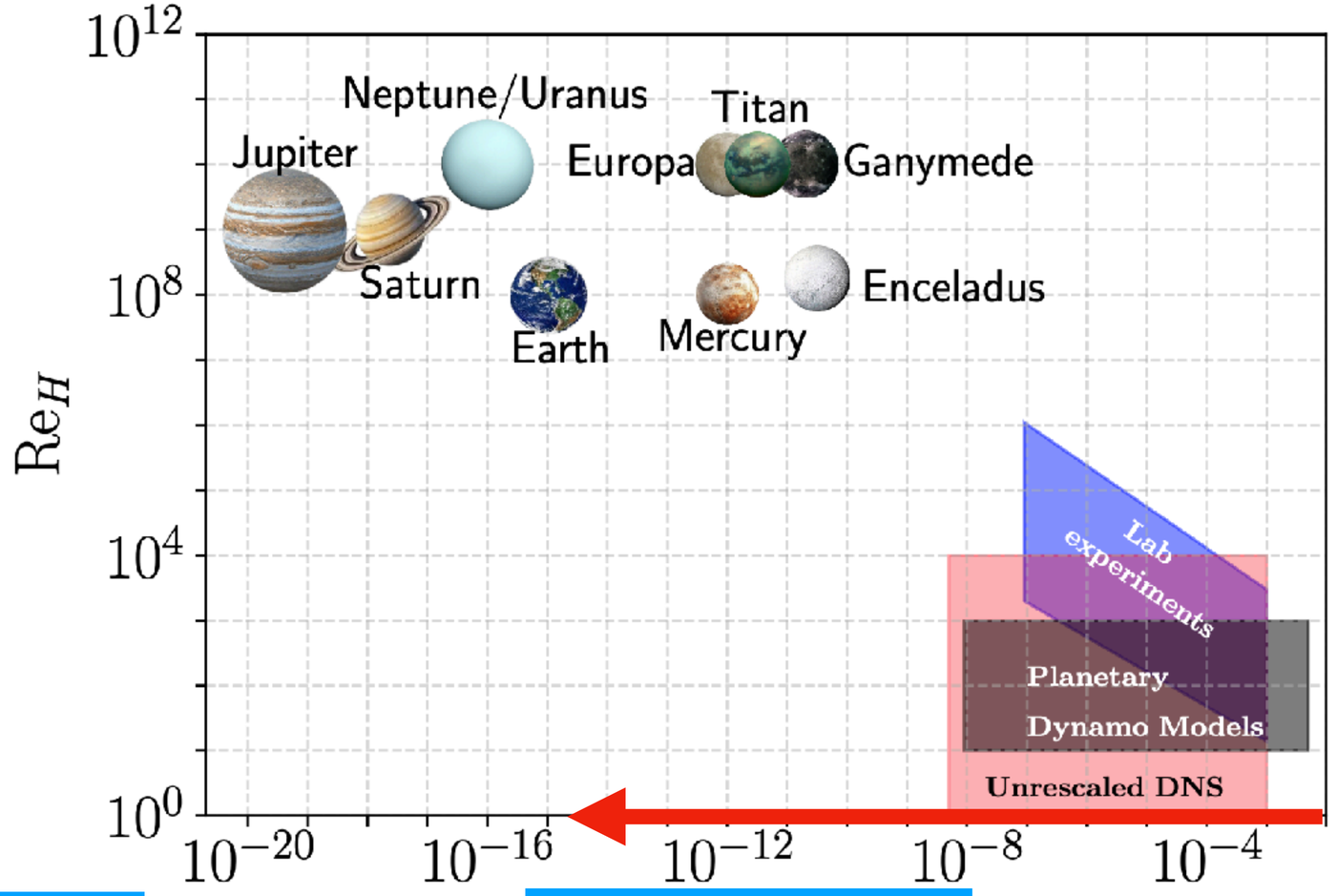
Time stepping: $\partial_t U = LU + N(U)$

Steady state solving: $0 = LU + N(U)$

Linear stability analysis: $\lambda u = Lu + N_U u$

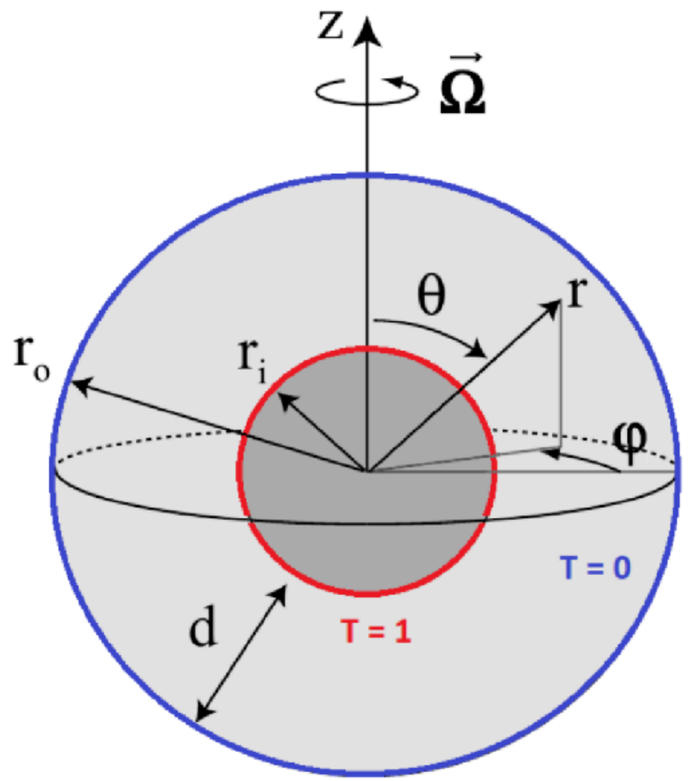
Rescaled Equations of Fluid Motion for Well-Conditioned Direct Numerical Simulations of Rapidly Rotating Convection

Keith Julien[1a], Adrian van Kan[2b], Benjamin Miquel[c], Edgar Knobloch[b], Geoffrey Vasil[d]

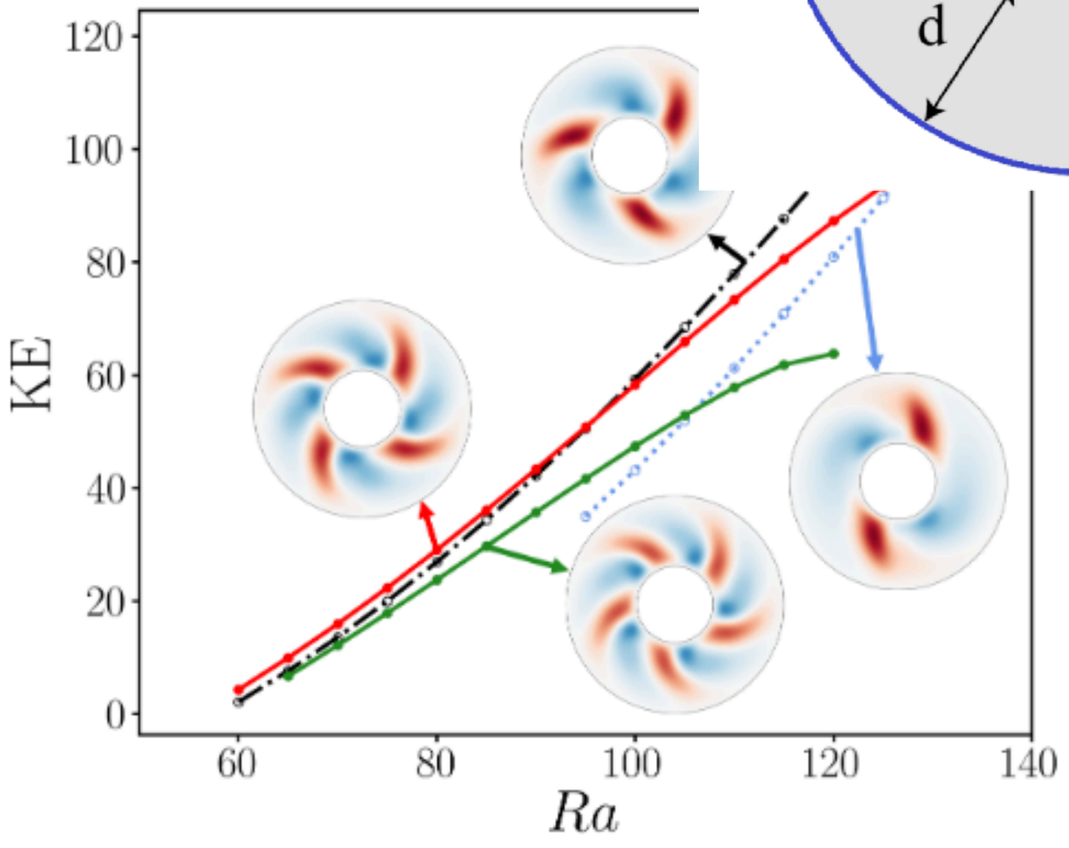


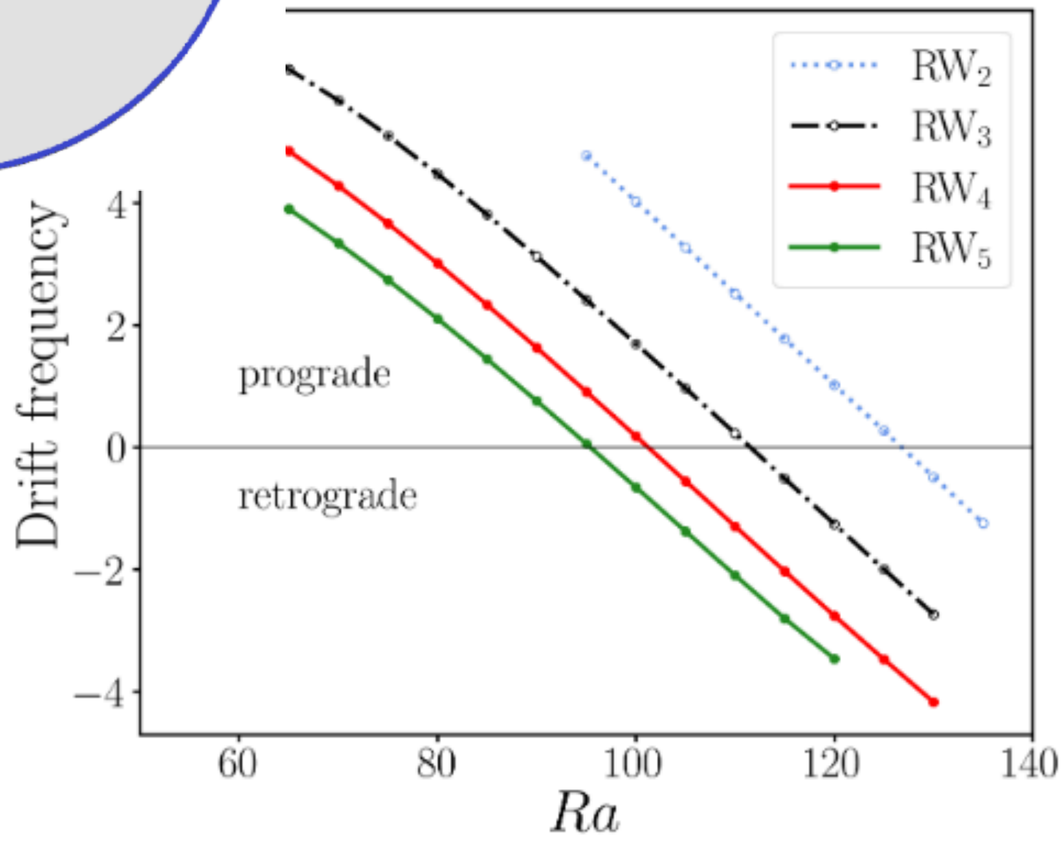
Challenge: go to low Ek

$$Ek \equiv \frac{\nu}{\Omega d^2}$$


z
$\vec{\Omega}$
θ
r
r_o
r_i
φ
T = 0
T = 1
d

$Ek = 10^{-3}$


KE
Ra
120
100
80
60
40
20
0
60
80
100
120
140


Drift frequency
Ra
RW2
RW3
RW4
RW5
prograde
retrograde
4
2
0
−2
−4
60
80
100
120
140

Heat Equation

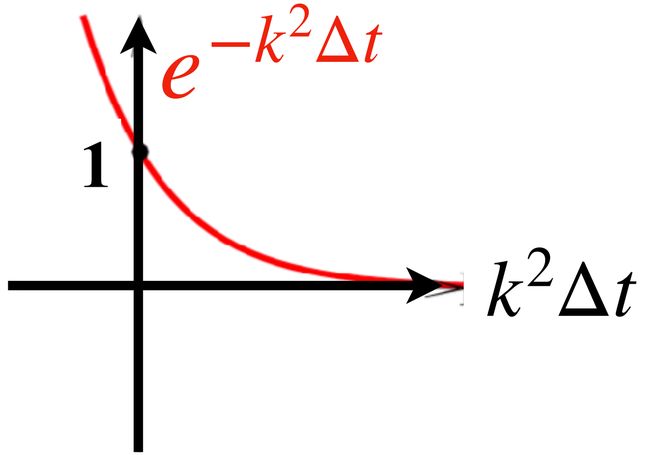
$$\partial_t u = \partial_{xx} u$$

$$u(x,t) = \sum_{k=1}^{k_{max}} \hat{u}_k(t) \sin kx$$

$$\partial_t \hat{u}_k = -k^2 \hat{u}_k$$

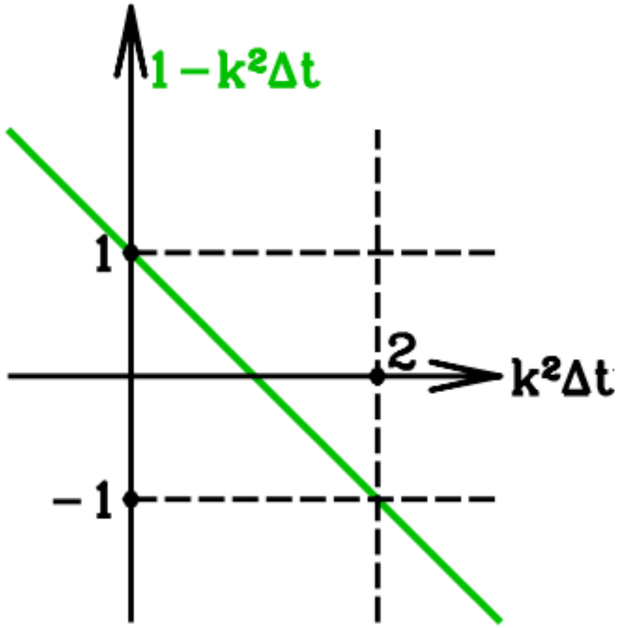
EXACT SOLUTION

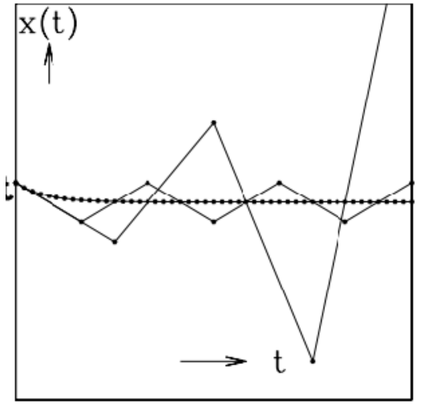
$$\hat{u}_k(t + \Delta t) = e^{-k^2 \Delta t} \hat{u}_k(t)$$



Explicit Euler

$$\begin{aligned}\hat{u}_k^{\mathrm{FE}}(t+\Delta t) &= \hat{u}_k^{\mathrm{FE}}(t) - k^2\Delta t\hat{u}_k^{\mathrm{FE}}(t) \\ &= (1-k^2\Delta t)\hat{u}_k^{\mathrm{FE}}(t)\end{aligned}$$



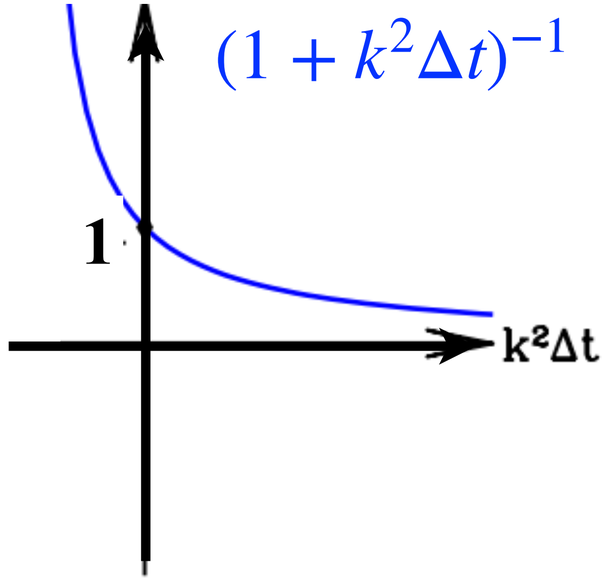


Implicit Euler

$$\begin{aligned}\hat{u}_k^{\mathrm{BE}}(t+\Delta t) &= \hat{u}_k^{\mathrm{BE}}(t) - k^2\Delta t\hat{u}_k^{\mathrm{BE}}(t+\Delta t) \\ (1+k^2\Delta t)\hat{u}_k^{\mathrm{BE}}(t+\Delta t) &= \hat{u}_k^{\mathrm{BE}}(t) \\ \hat{u}_k^{\mathrm{BE}}(t+\Delta t) &= (1+k^2\Delta t)^{-1}\hat{u}_k^{\mathrm{BE}}(t)\end{aligned}$$

Matrix version:

$$u^{\mathrm{BE}}(t+\Delta t) = (I-\Delta t L)^{-1}u^{\mathrm{BE}}(t)$$



Explicit methods approximate exponential by a polynomial so they will ALWAYS diverge for sufficiently large Δt

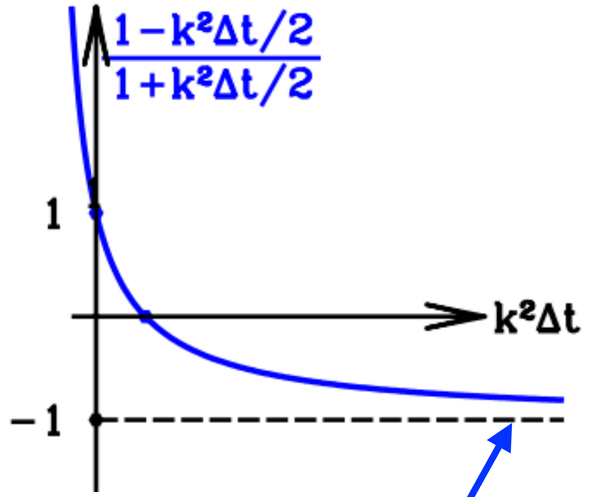
Implicit methods approximate exponential by a rational function.

Some second-order implicit methods

Crank-Nicolson (implicit) also called trapezoidal

$$u^{\mathrm{CN}}(t+\Delta t) = u^{\mathrm{CN}}(t)+\Delta t\left(\frac{1}{2}f(u^{\mathrm{CN}}(t)) + \frac{1}{2}f(u^{\mathrm{CN}}(t+\Delta t))\right.$$

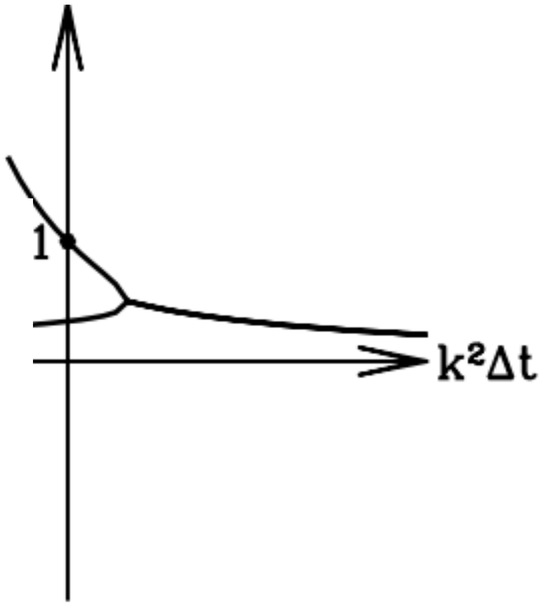
$$\hat{u}_k^{\mathrm{CN}}(t+\Delta t) = \frac{1-\frac{k^2\Delta t}{2}}{1+\frac{k^2\Delta t}{2}}\quad \hat{u}_k^{\mathrm{CN}}(t)$$



|Amp. factor| < 1 but spurious large-k behavior: slow oscillatory decay

Backwards Differentiation (implicit)

$$u^{\mathrm{BD}}(t+\Delta t) = \frac{4}{3}u^{\mathrm{BD}}(t)-\frac{1}{3}u^{\mathrm{BD}}(t-\Delta t)+\frac{2}{3}\Delta t f(u^{\mathrm{BD}}(t+\Delta t))$$



Write spherical rotating convection code with implicit Coriolis timestepping

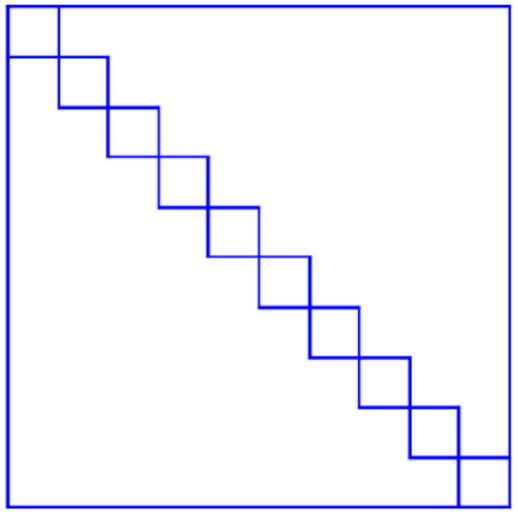
Riquier → Rambert → Gonzalez Sembla



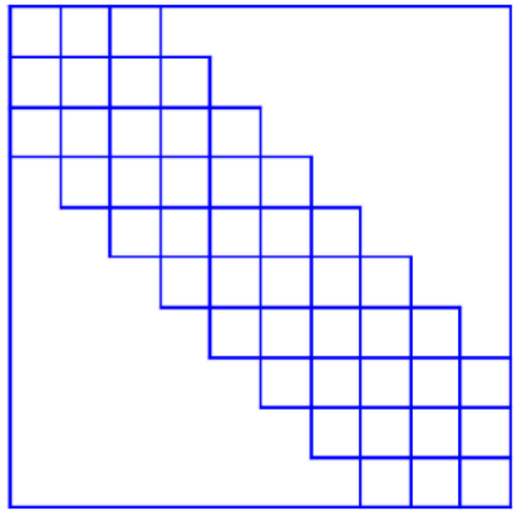


$$\frac{\partial \mathbf{U}}{\partial t} + (\mathbf{U}\cdot\nabla)\,\mathbf{U} - \nabla^2\mathbf{U} = \frac{1}{Ek}\left[-\nabla P + Ra\,T\mathbf{r} - 2\mathbf{e}_z\times\mathbf{U}\right]$$

explicit implicit absorb implicit



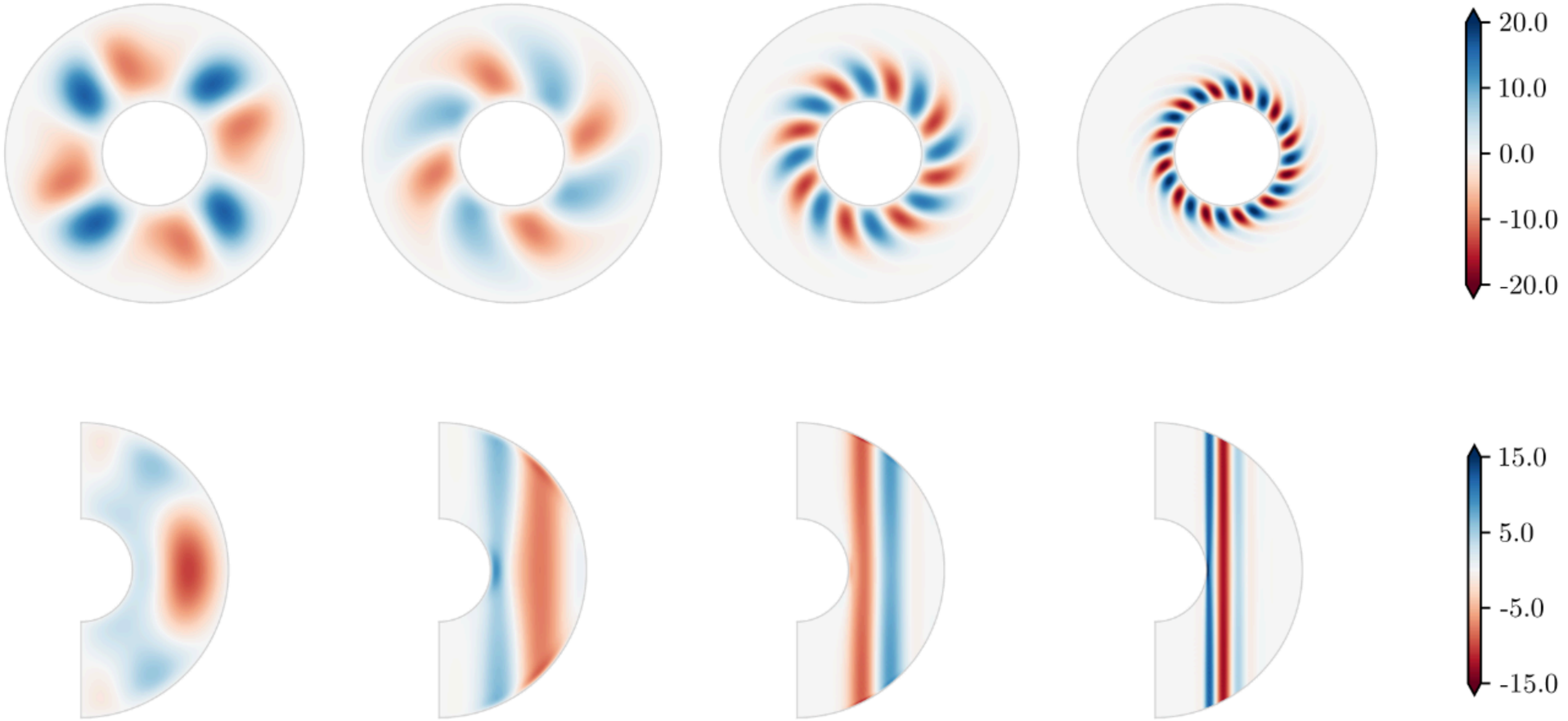
Block diagonal matrix



Block pentadiagonal matrix

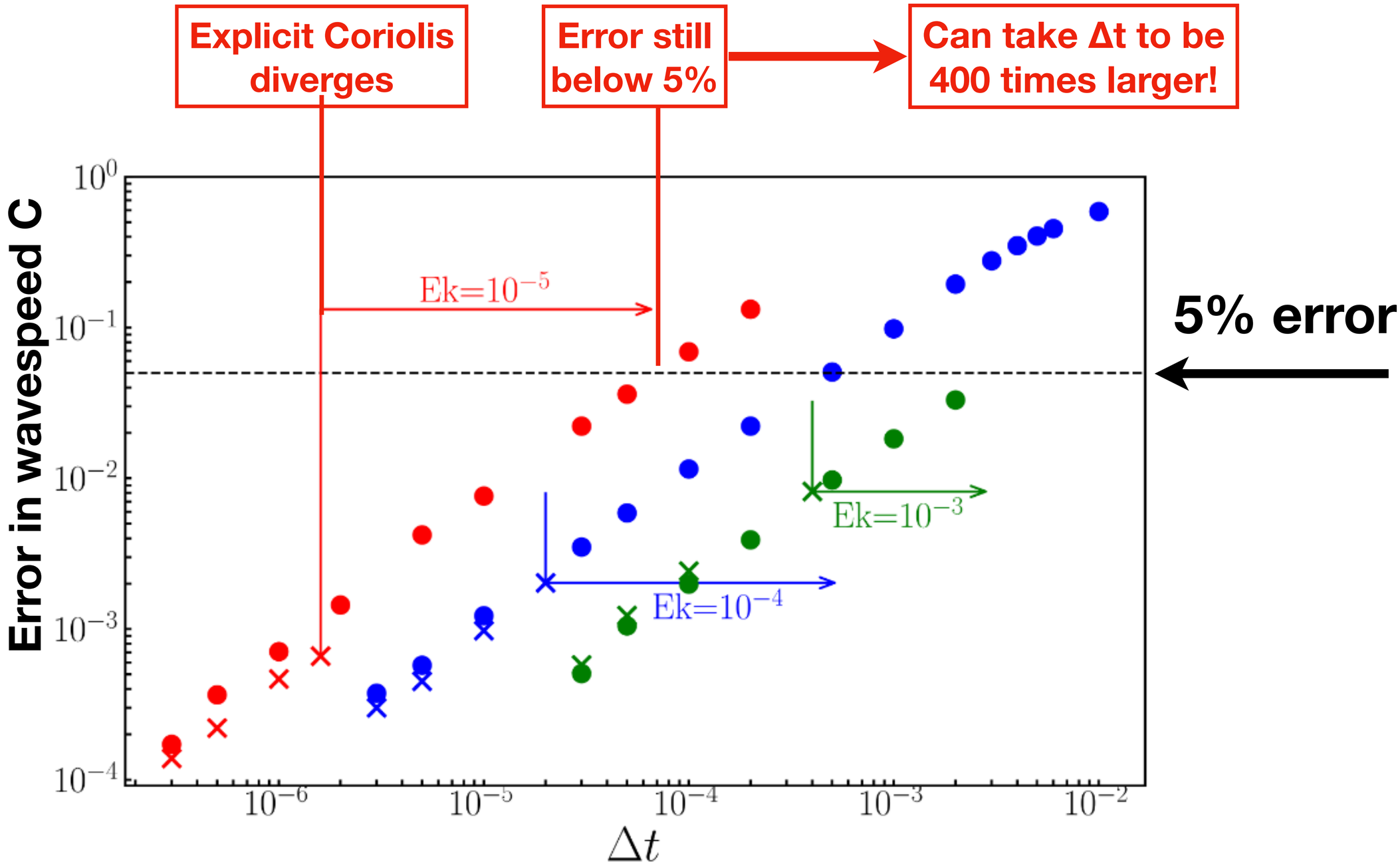
because Coriolis term couples spherical harmonics Y_ℓ^m

$Ek = 10^{-2}$ $Ek = 10^{-3}$ $Ek = 10^{-4}$ $Ek = 10^{-5}$



$M \sim Ek^{-1/3}$

$(r_{\max} - r_{\text{in}}) \sim Ek^{2/9}$


Explicit Coriolis diverges
Error still below 5%
Can take Δt to be 400 times larger!
5% error
Error in wavespeed C
10^{0}
10^{-1}
10^{-2}
10^{-3}
10^{-4}
10^{-6}
10^{-5}
10^{-4}
10^{-3}
10^{-2}
Δt
Ek=10^{-5}
Ek=10^{-4}
Ek=10^{-3}

Newton's method for computing steady states and traveling waves

starting from a timestepping code

Navier-Stokes

$$\frac{\partial U}{\partial t} = LU + N(U)$$

FEBE: Forward Euler for Nonlinear, Backward Euler for Linear:

$$B_{\Delta t} \equiv (I - \Delta t L)^{-1}(I + \Delta t N)$$

Steady states

$$0 = LU + N(U)$$

Why use backward (implicit) Euler for L and forward (explicit) Euler for N?

Because L ≈ Laplacian ≈ -k² is responsible for large range of time scales (≈ stiffness)

→ large range of eigenvalues (≈ poor conditioning)

Fortunately, it is also easier to invert L (linear) than N (nonlinear)

$$B_{\Delta t} - I = (I - \Delta t L)^{-1}(I + \Delta t N) - I$$

$$= (I - \Delta t L)^{-1}\left[(I + \Delta t N) - (I - \Delta t L)\right]$$

$$= (I - \Delta t L)^{-1}\Delta t(L + N)$$ **has same roots as *L+N***

$$\begin{cases} \approx \Delta t(L+N) & \text{for } \Delta t \text{ small} \\ \approx -L^{-1}(L+N) & \text{for } \Delta t \text{ large} \end{cases}$$

Hence Δ*t* interpolates between

$$\begin{cases} \text{no preconditioner} & \text{for } \Delta t \text{ small} \\ \text{preconditioning by } L^{-1} & \text{for } \Delta t \text{ large} \end{cases}$$

Note that these operations are only algebraic:

We do NOT rely here on $B_{\Delta t} \approx \exp(\Delta t\,(L+N))$ for $\Delta t \ll 1$ and so can take Δt large

Find roots via Newton's method

$$DF_U\, u = F(U) \qquad U \leftarrow U - u$$

$$(L + N_U)\, u = (L + N)\, U$$

$$\iff (I - \Delta t L)^{-1} \Delta t (N_U + L)\, u = (I - \Delta t L)^{-1} \Delta t (N + L)\, U$$

$$\iff [(I - \Delta t L)^{-1}(I + \Delta t N_U) - I]\, u = [(I - \Delta t L)^{-1}(I + \Delta t N) - I]\, U$$

difference between two long consecutive linearized timesteps

difference between two long consecutive timesteps

Solve linear systems via BiCGSTAB, GMRES, or IDR

Traveling waves: $U(\phi - Ct)$

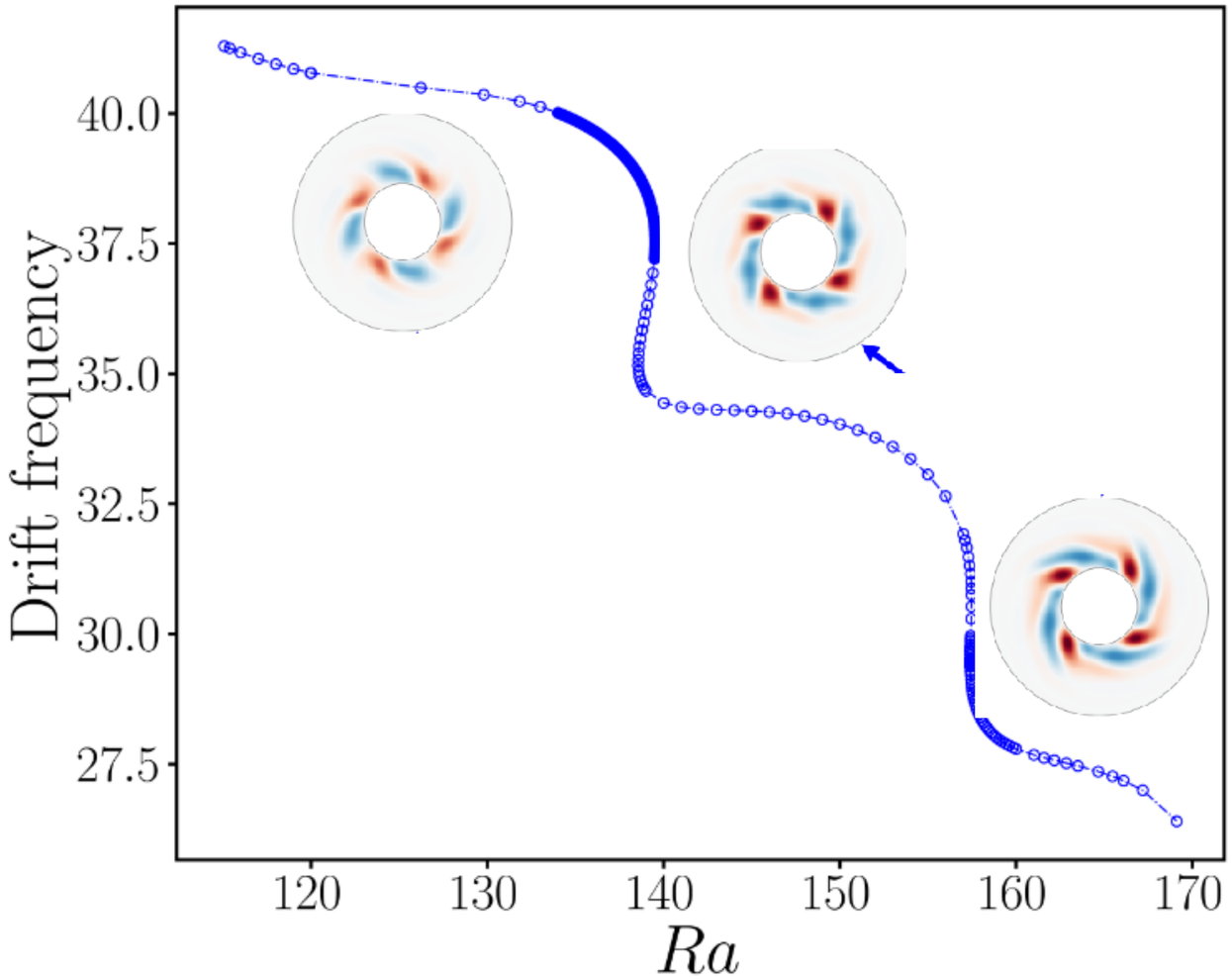
$$\text{Goal}: \begin{cases} 0 = C\partial_\phi U + N(U) + LU \\ 0 = p(U) - \bar{p} \end{cases}$$

$$\left[\begin{array}{c|c} C\partial_\phi + N_U + L & \partial_\phi U \\ \hline 0\ 0 \ldots 0\ 1\ 0 \ldots 0 & 0 \end{array}\right] \begin{bmatrix} u \\ c \end{bmatrix} = \begin{bmatrix} C\partial_\phi U + N(U) + LU \\ U_i - \bar{p} \end{bmatrix}$$

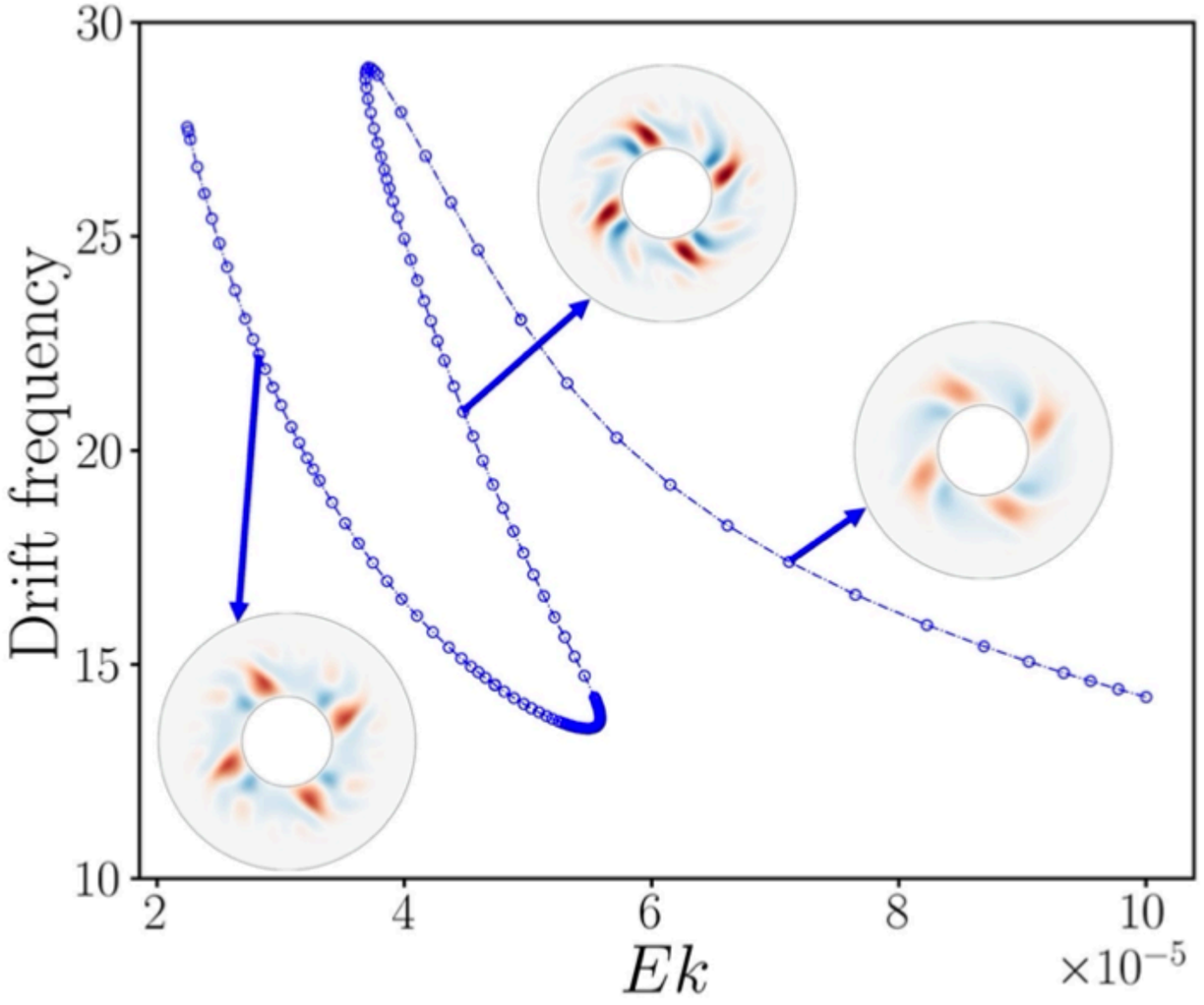
Trick: use trivial phase condition, store (C,c) in corresponding element of (U,u)

Branch following via Newton's method: going around saddle-node bifurcations

$Ek = 3.53 \times 10^{-5}$



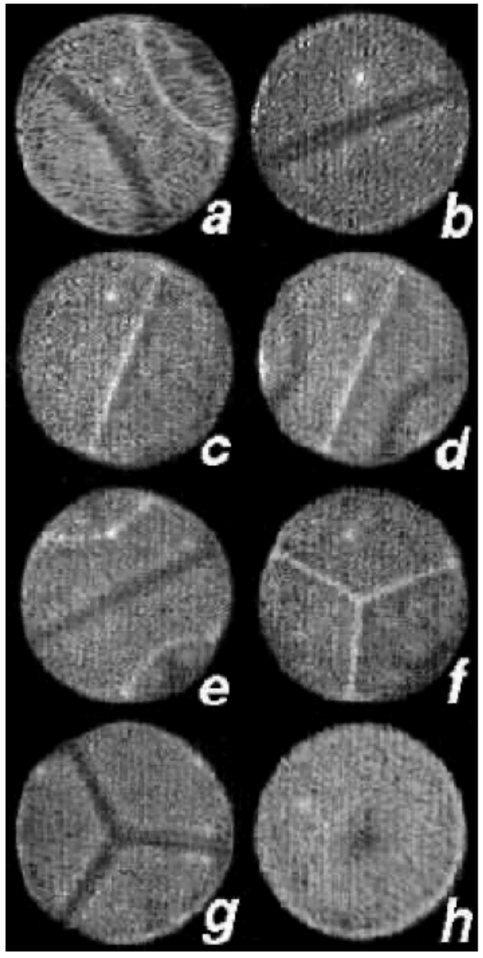
$Ra = 6.5 \times Ek^{-1/3}$

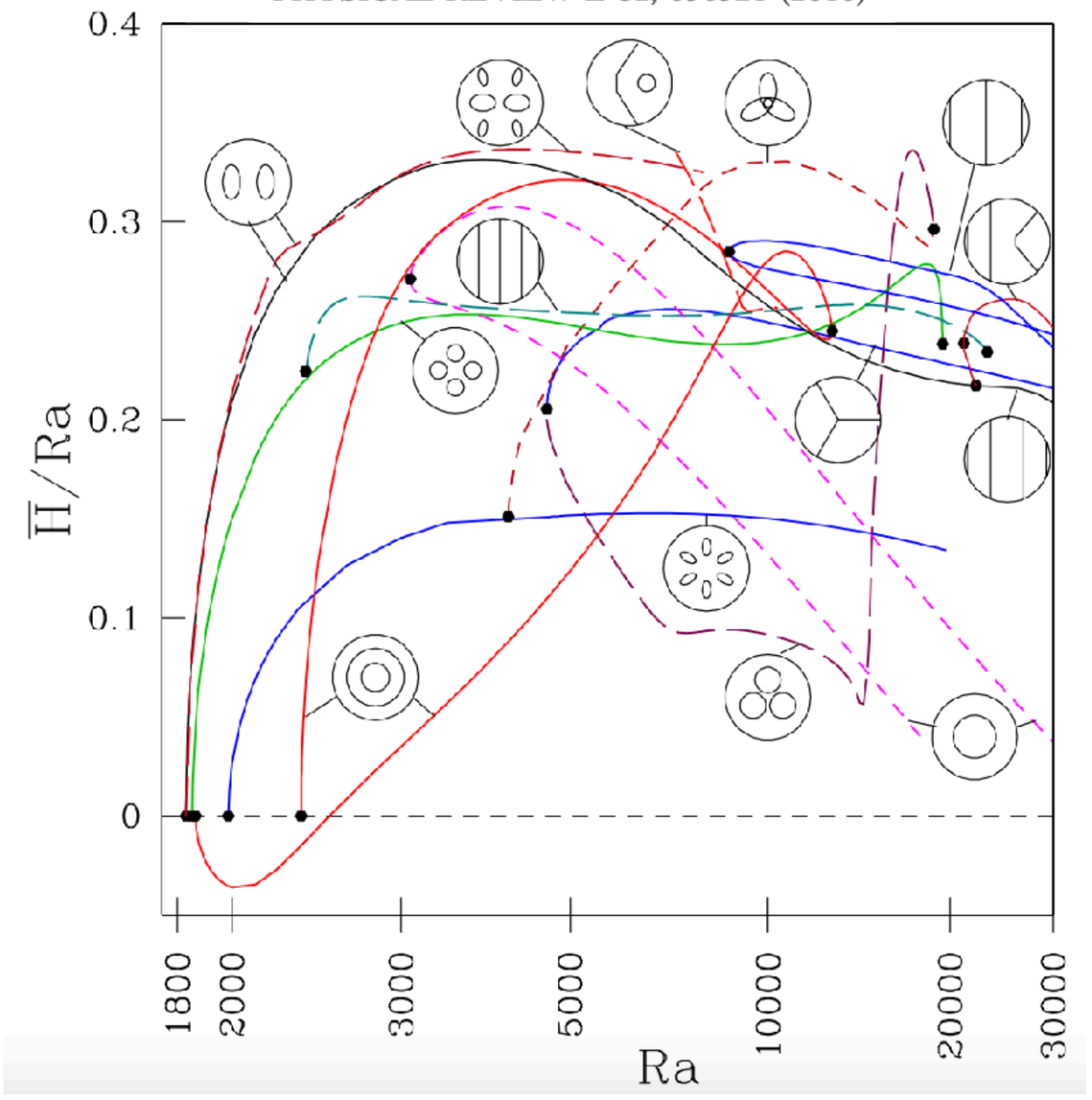


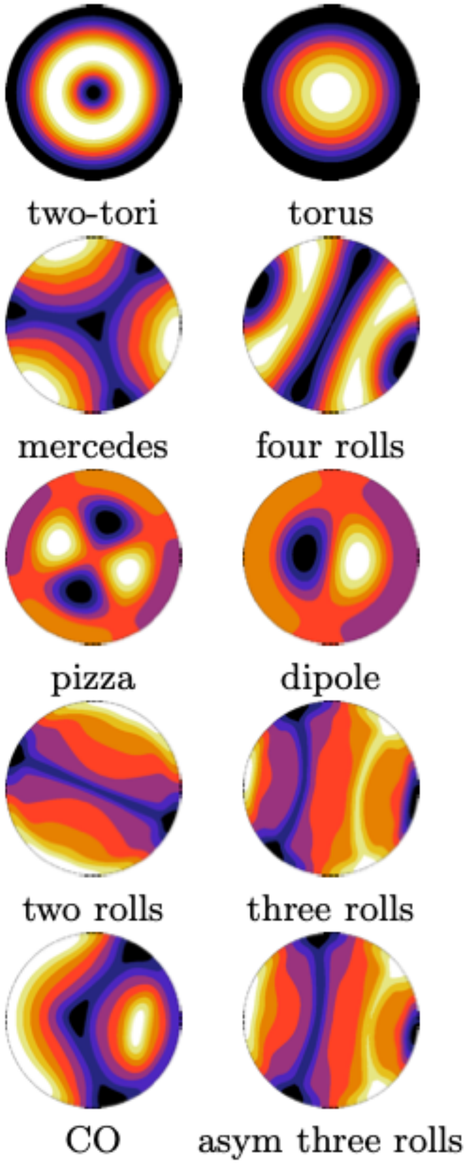
The system LOVES some drift frequencies and HATES others!

Automatic change in numerical resolution along branch!

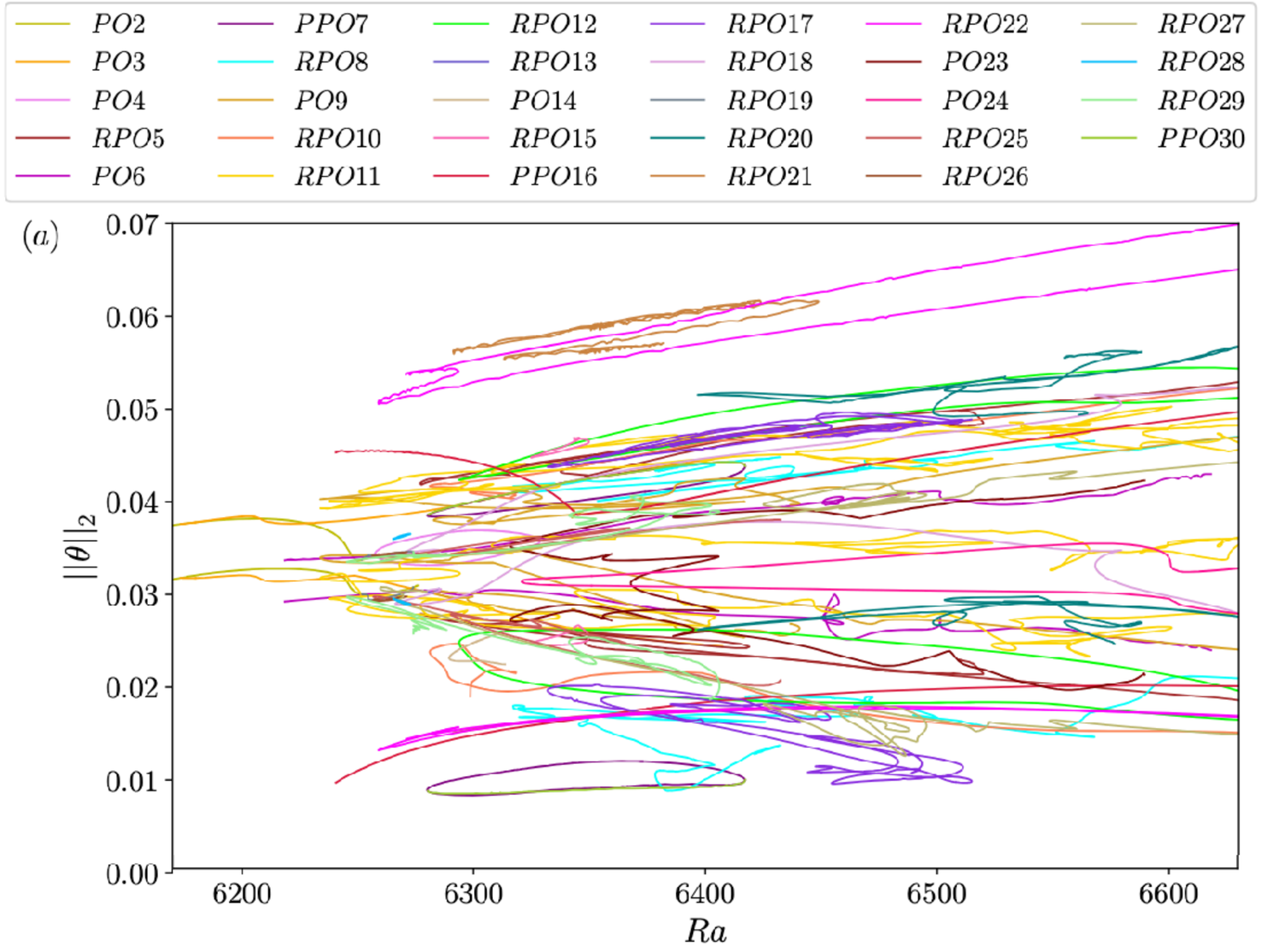
PHYSICAL REVIEW E **81**, 036321 (2010)

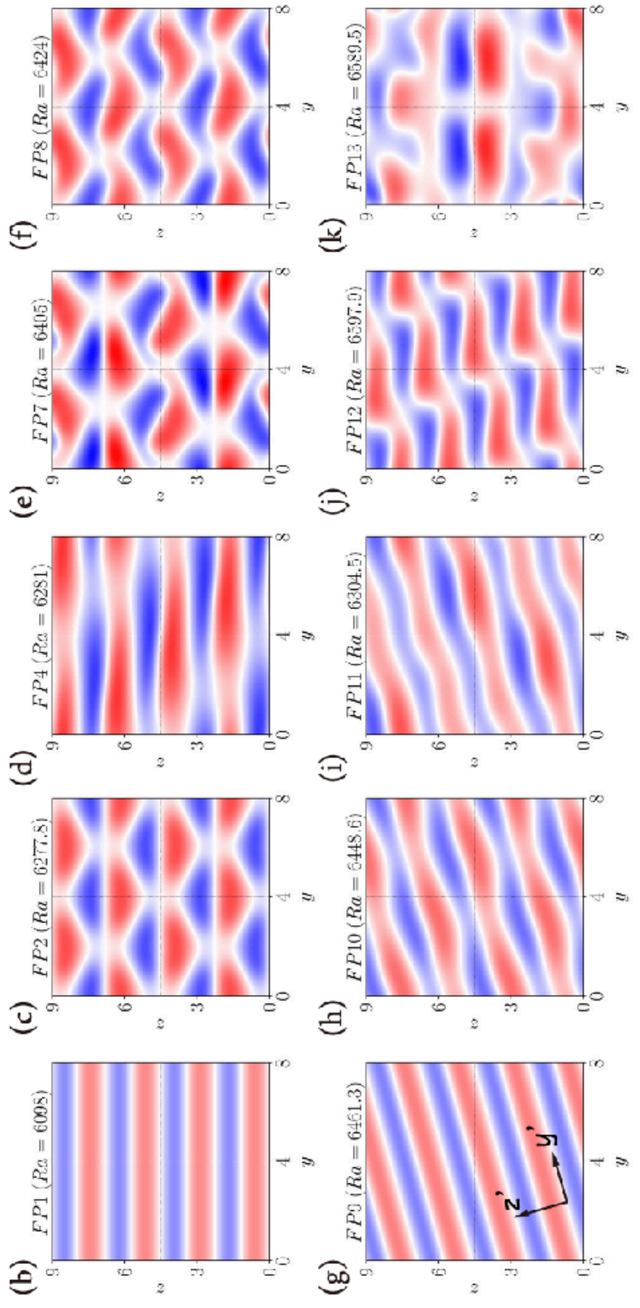






Hof, Lucas & Mullin, Phys. Fluids (1999)


PO2
PO3
PO4
RPO5
PO6
PPO7
RPO8
PO9
RPO10
RPO11
RPO12
RPO13
PO14
RPO15
PPO16
RPO17
RPO18
RPO19
RPO20
RPO21
RPO22
PO23
PO24
RPO25
RPO26
RPO27
RPO28
RPO29
PPO30
(a)
0.07
0.06
0.05
0.04
0.03
0.02
0.01
0.00
$||\theta||_2$
6200
6300
6400
6500
6600
Ra


(f)
FP8 (Ra = 6424)
(k)
FP13 (Ra = 6589.5)
(e)
FP7 (Ra = 6405)
(j)
FP12 (Ra = 6597.9)
(d)
FP4 (Ra = 6281)
(i)
FP11 (Ra = 6304.5)
(c)
FP2 (Ra = 6277.8)
(h)
FP10 (Ra = 6448.6)
(b)
FP1 (Ra = 6098)
(g)
FP3 (Ra = 6451.3)
z'
y'
y

Linear Stability Analysis

$$\lambda u = Lu + N_U u$$

How to calculate eigenpairs (λ, u)?

1) Direct: Diagonalisation = QR decomposition

Storage: M^2 **Time: M^3**

For 3D case with $M_x = M_y = M_z = 10^2$, we have $M = 10^6$

$M^2 = 10^{12}$ $M^3 = 10^{18}$

2) Iterative: Calculate a few desired eigenpairs.

Use only matrix-vector products $u \to Au$

To diagonalise an arbitrary matrix,

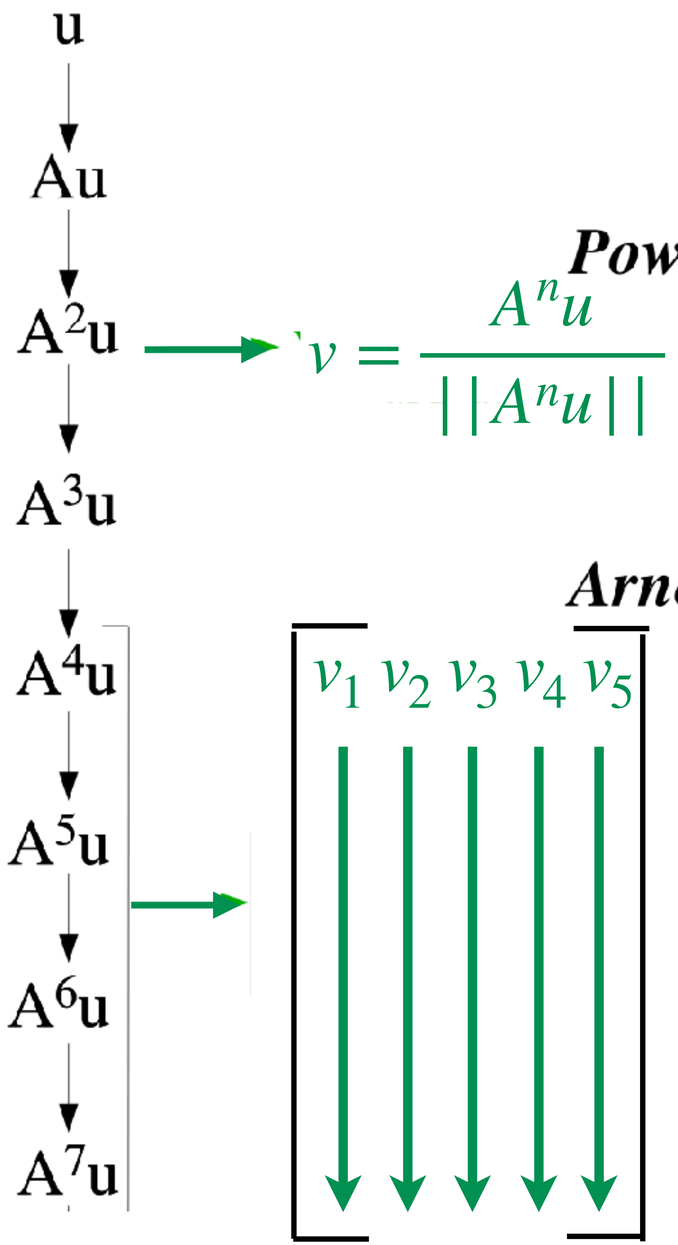
Each product $u \to Au$ requires M^2 operations
Generating M eigenpairs requires M iterations } M^3

Can gain:

If A is structured or sparse, then $u \to Au$ takes $\sim M$ ops.

Aim method at desired eigenvalues.

Leading Eigenvalues: $A\Psi = \lambda\Psi$



Power Method

$$v = \frac{A^n u}{||A^n u||}$$

$\lambda \approx \langle v, Av \rangle$ $\quad \Psi \approx v$

Arnoldi Decomposition

$H = V^T A V$ $\quad H_{ij} \equiv \langle v_i, Av_j \rangle$

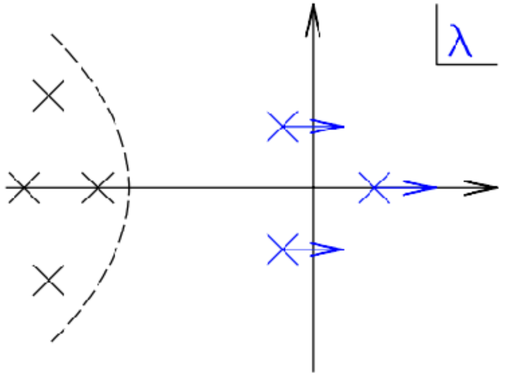
$K \times K \quad K \times N \quad N \times N \quad N \times K$

$A \approx V H V^T$ $\quad AV \approx VH$

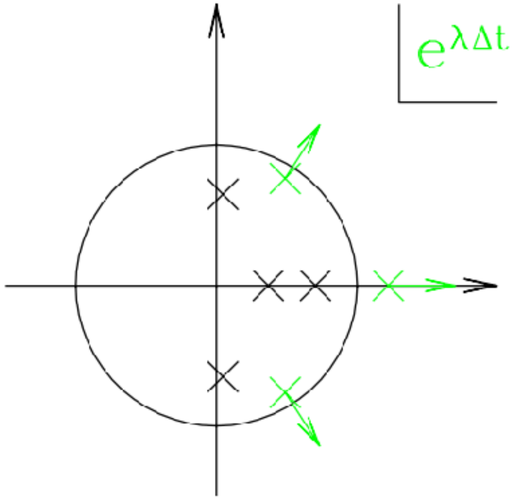
$N \times N \quad N \times K \quad K \times K \quad K \times N$

$H\phi = \lambda\phi$ $\quad AV\phi \approx VH\phi = \lambda V\phi$

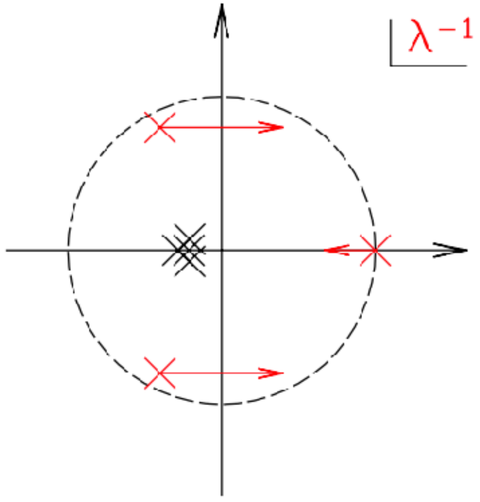
MATRIX TRANSFORMATIONS



$$\text{If } A\,u = \lambda\,u$$
$$\text{then } f(A)\,u = f(\lambda)\,u$$



$$f(A) = e^{A\Delta t}$$



$$f(A) = A^{-1}$$

$f(A) = \sum_j f_j A^j$

f_j chosen dynamically to extract desired eigenvalues: principle of ARPACK (Sorensen et al.)

Exponential Power Method

$$u_{n+1} = (I - \Delta t L)^{-1}(I + \Delta t N_U)u_n \approx e^{\Delta t(L+N_U)}u_n$$

Approximation valid for $\Delta t \ll 1$

Time-stepping linearized evolution equation

Enhancement factor at each iteration is

$$\left|\frac{e^{\Delta t \lambda_1}}{e^{\Delta t \lambda_2}}\right| \gtrsim 1 \qquad \text{where } \lambda_1 > \lambda_2 > \cdots$$

Inverse Power Method

$$u_{n+1} = (L + N_U)^{-1} u_n$$

Stokes preconditioning:

$$\begin{aligned}
(L + N_U) u_{n+1} &= u_n \\
(I - \Delta t L)^{-1} \Delta t (L + N_U) u_{n+1} &= (I - \Delta t L)^{-1} \Delta t u_n \\
(I - \Delta t L)^{-1} \left[I + \Delta t N_U - (I - \Delta t L) \right] u_{n+1} &= (I - \Delta t L)^{-1} \Delta t u_n \\
\underbrace{\left[(I - \Delta t L)^{-1} (I + \Delta t N_U) - I \right] u_{n+1}}_{\text{difference between two widely spaced consecutive linearized timesteps}} &= \underbrace{(I - \Delta t L)^{-1} \Delta t u_n}_{\text{one Stokes timestep}}
\end{aligned}$$

Solve with Conjugate Gradient (Bi-CGSTAB) method.

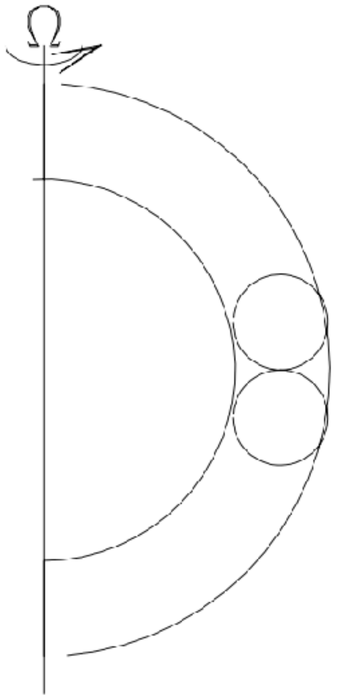
Enhancement factor at each iteration is $\left| \frac{\lambda_2}{\lambda_1} \right| \gg 1$ **for** $\lambda_1 \approx 0$

Can shift to find eigenvalues closest to s

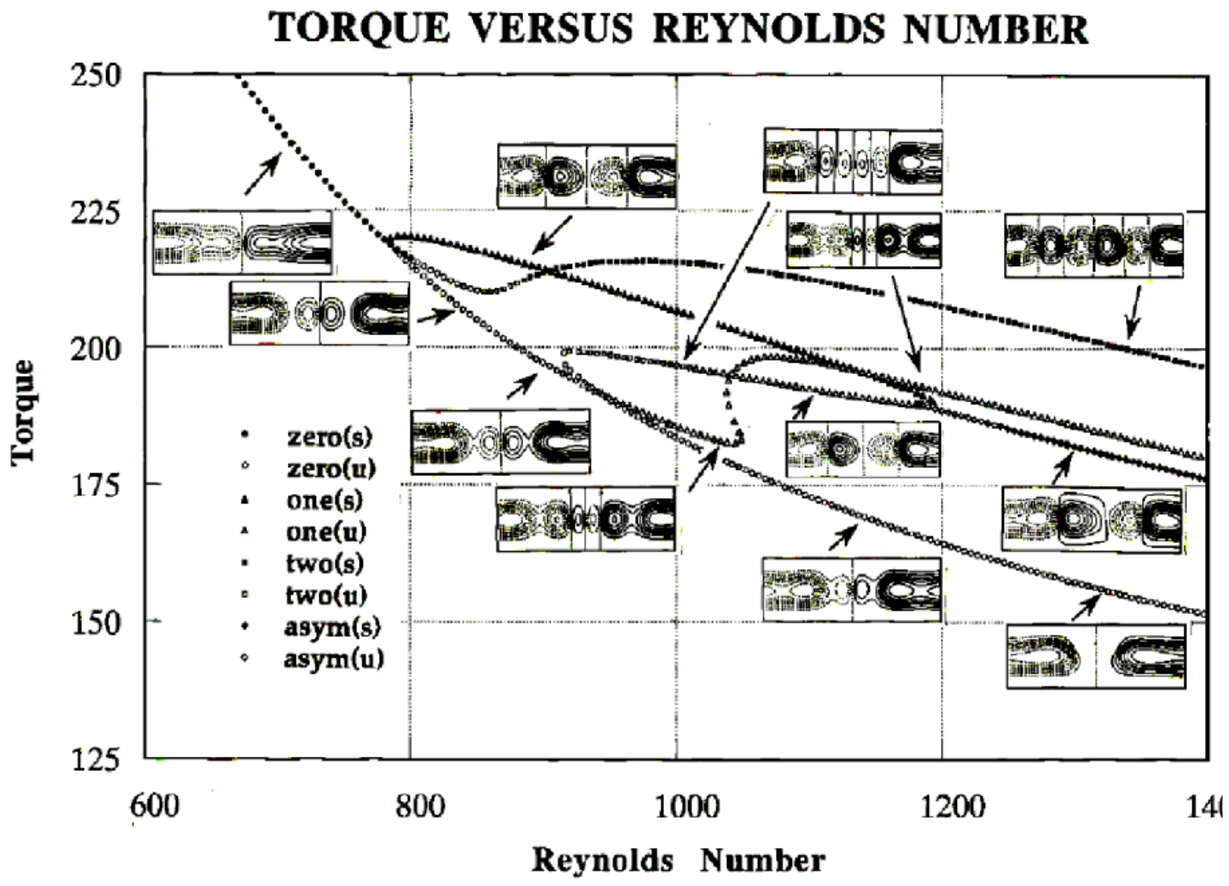
$$\left| \frac{\lambda_2 - s}{\lambda_1 - s} \right| \gg 1 \qquad \text{for } \lambda_1 \approx s$$

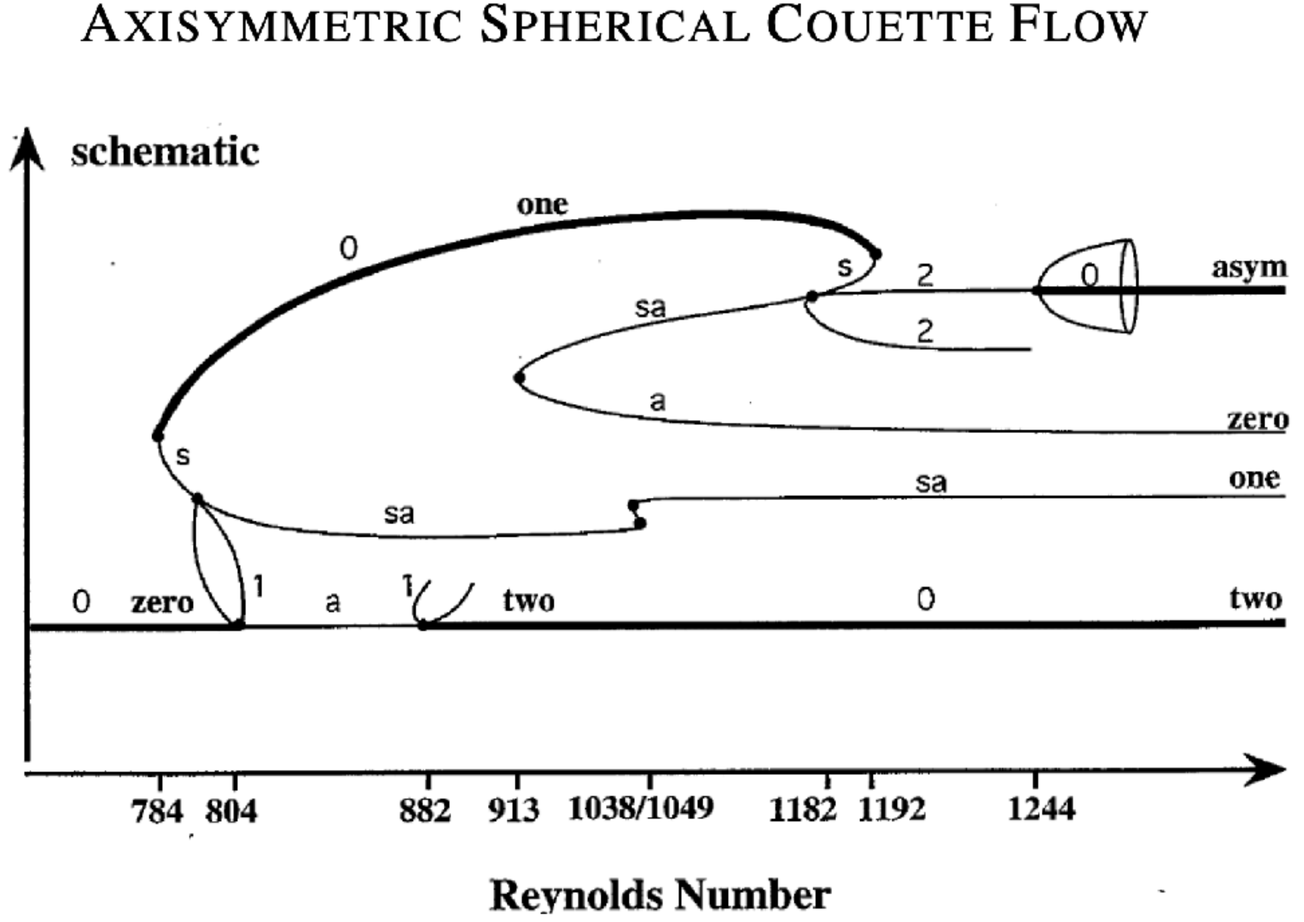
Axisymmetric Spherical Couette Flow with $(r_2 - r_1)/r_1 = 0.18$

with C.K. Mamun

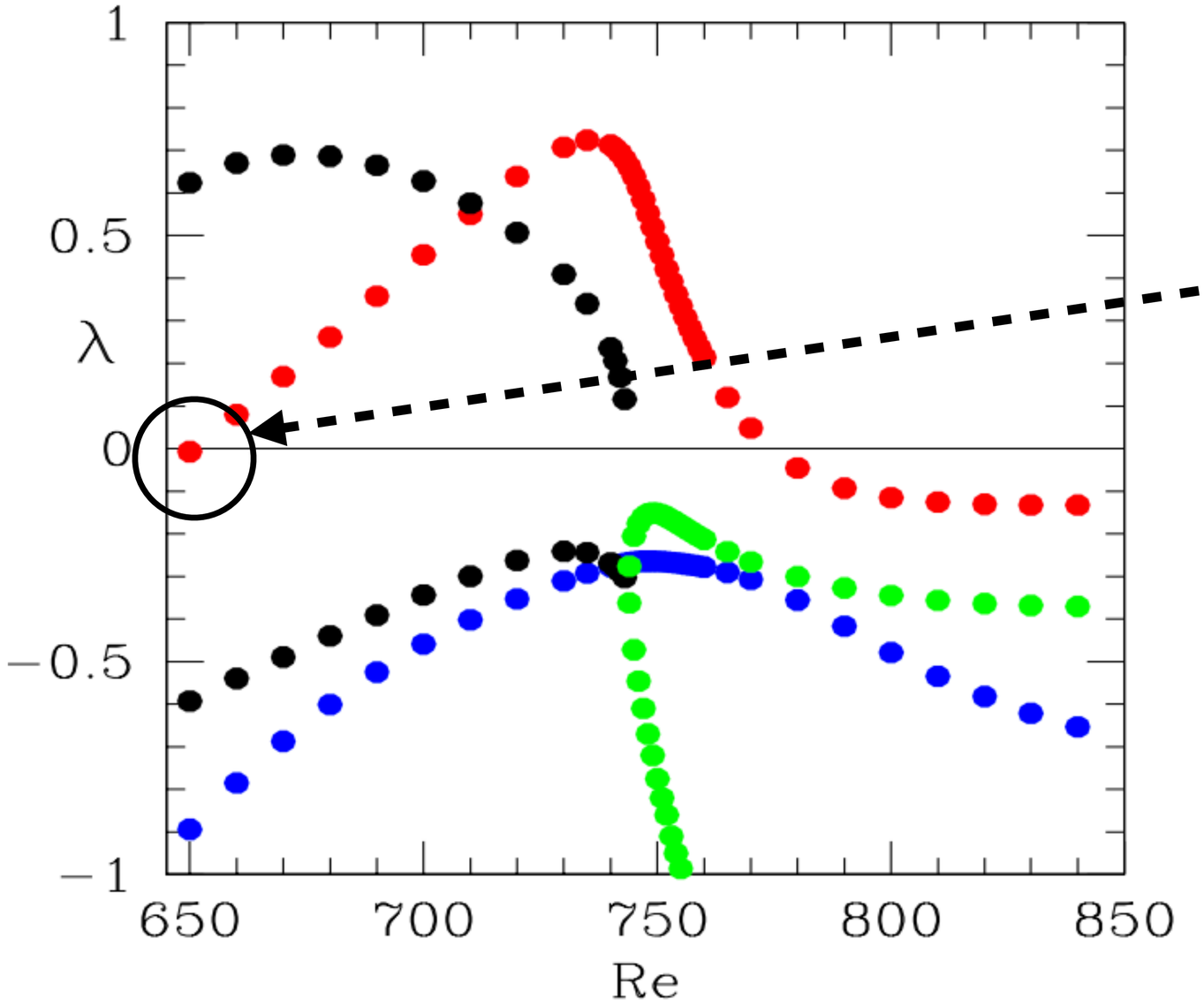


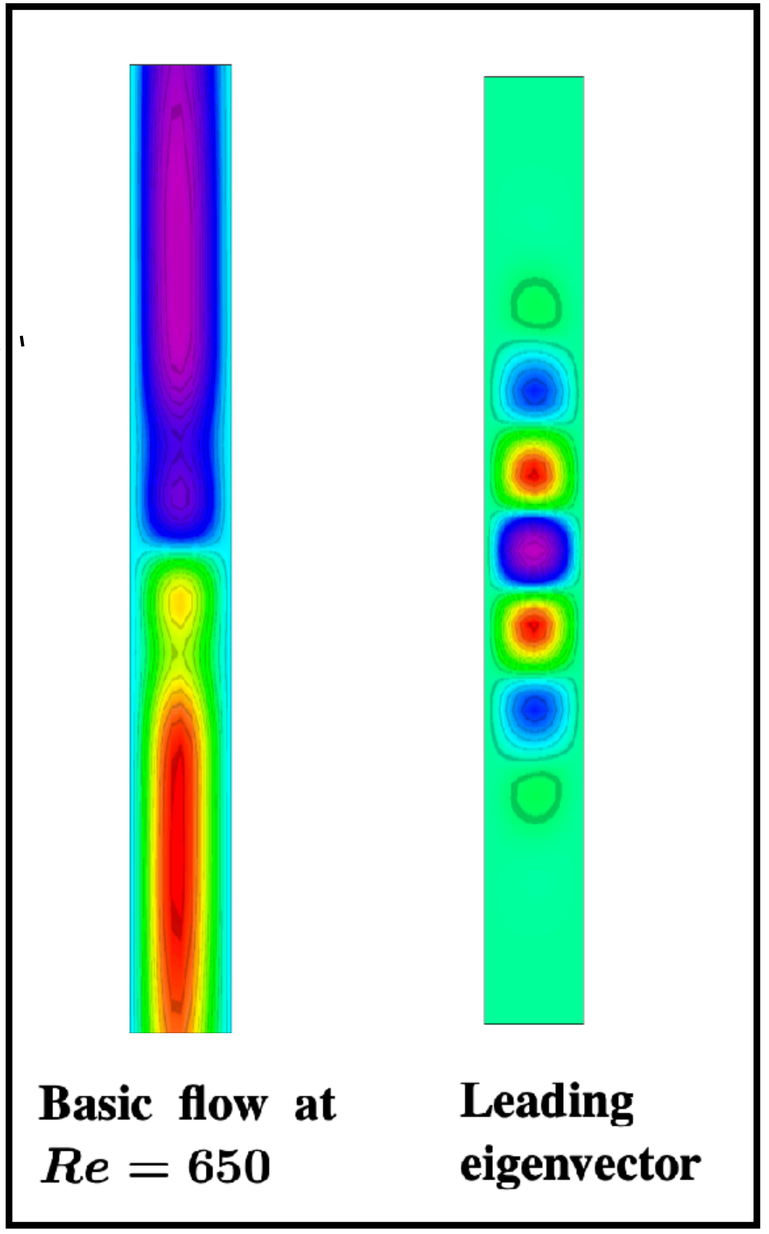
straighten out the diagram



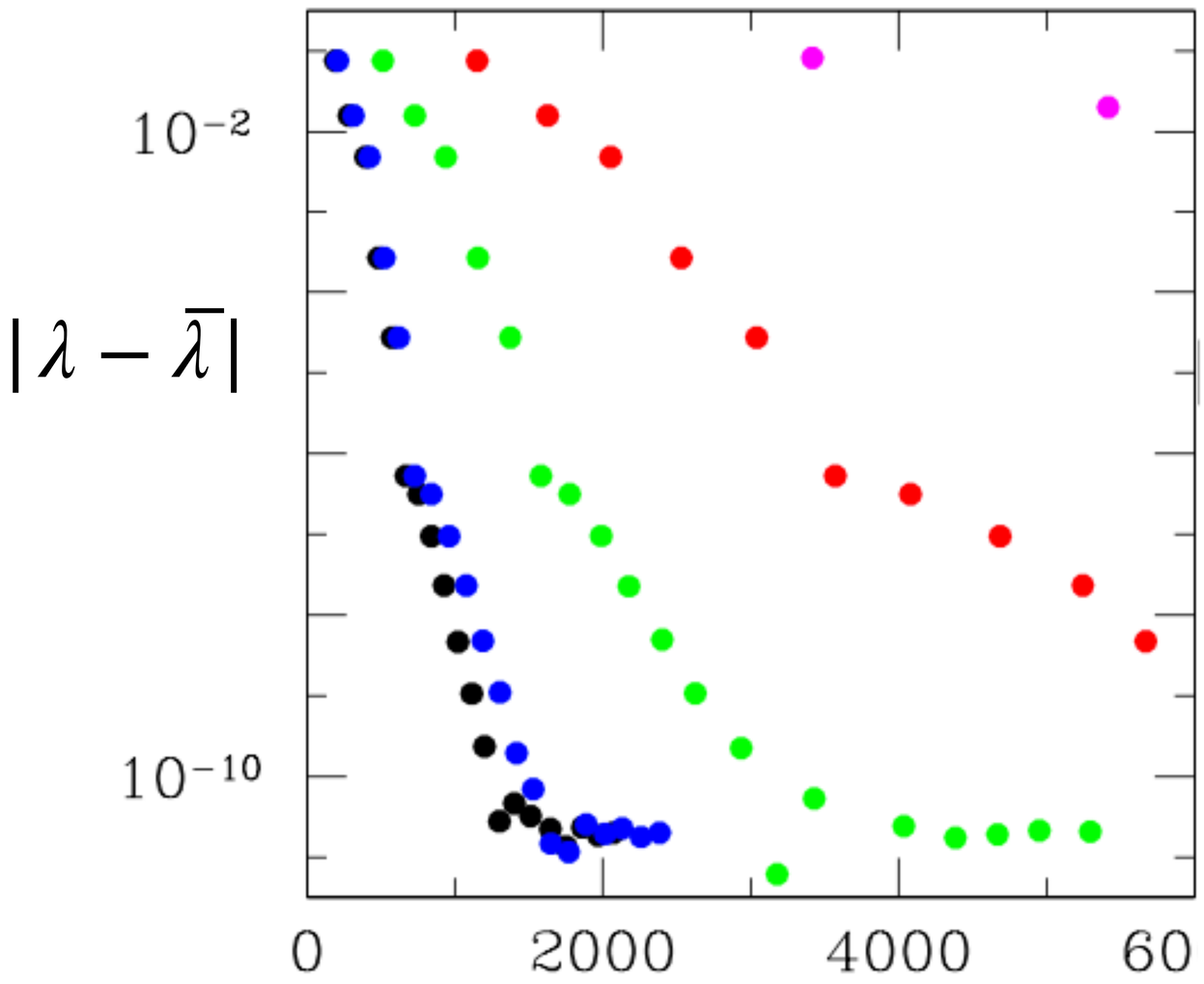


Inverse Power Method on Spherical Couette Flow



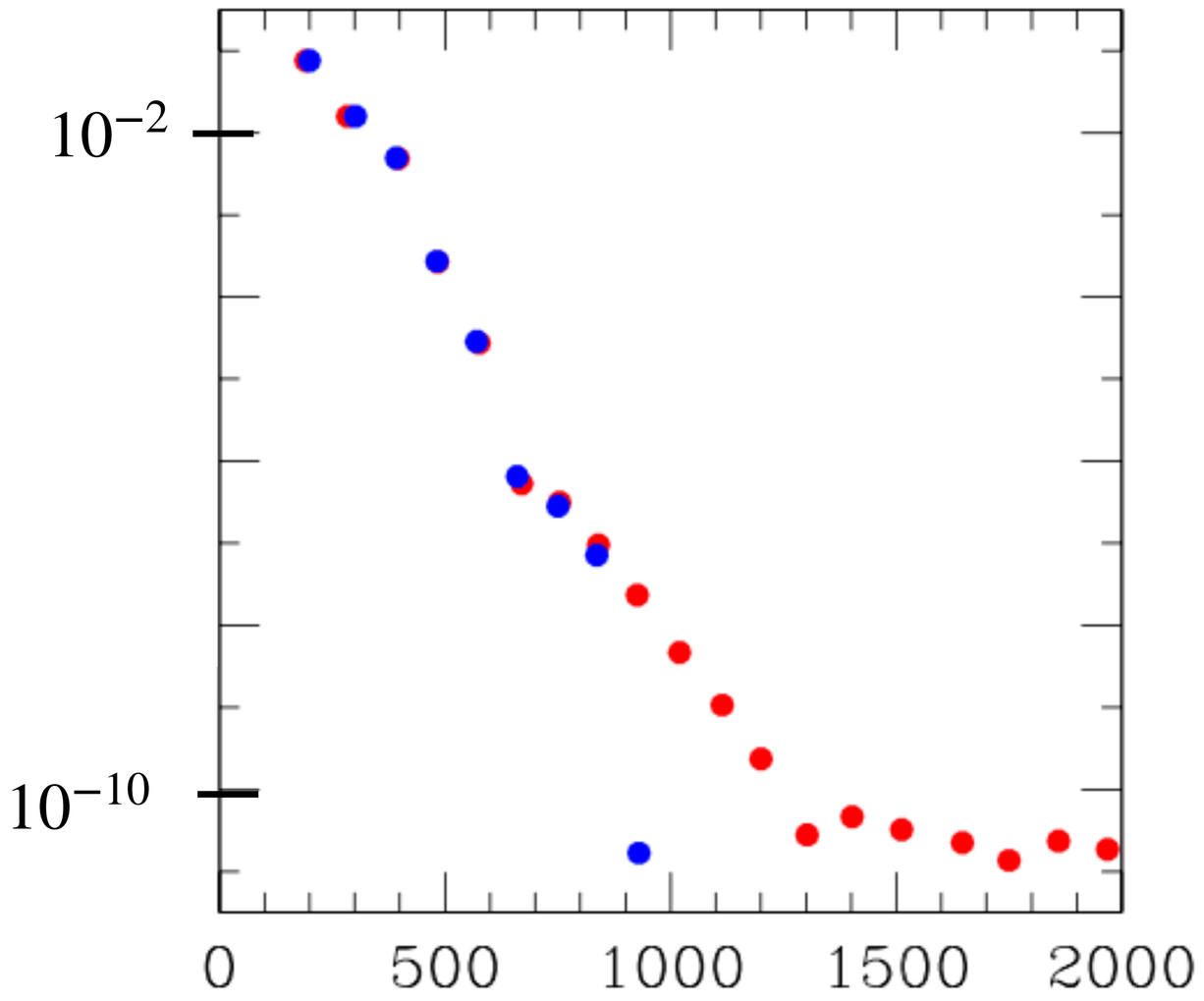


Δt =100, 10, 1, 0.1, 0.01



linearized timesteps
(operator actions)

$N_r \times N_\theta$ = 4096, 16384



linearized timesteps
(operator actions)

Optimal Forcing

with Mattias Brynjell-Rahkola, Philipp Schlatter, Dan Henningson, KTH



$$
\begin{aligned}
\partial_t \mathbf{u}(\mathbf{x},t) &= \mathcal{A}\mathbf{u}(\mathbf{x},t) + \mathbf{f}(\mathbf{x})e^{i\omega t} \\
u(\mathbf{x},t) &= -(\mathcal{A}-i\omega I)^{-1}\mathbf{f}(\mathbf{x})e^{i\omega t} + e^{\mathcal{A}t}c(\mathbf{x}) \\
&\Longrightarrow -(\mathcal{A}-i\omega I)^{-1}\mathbf{f}(\mathbf{x})e^{i\omega t} \qquad \text{if all eigs of } \mathcal{A} \text{ are negative} \\
&\equiv -\mathcal{R}(i\omega)\mathbf{f}(\mathbf{x})e^{i\omega t}
\end{aligned}
$$

Seek profile $\mathbf{f}(\mathbf{x})$ and frequency ω which yields maximum

$$G(\omega) = \max_{\mathbf{f}(x)} \frac{||\mathcal{R}(i\omega)\mathbf{f}||}{||\mathbf{f}||}$$

This is the maximum eigenvalue of

$$\mathcal{R}(i\omega)\mathcal{R}^{\dagger}(i\omega) = \left((\mathcal{A}-i\omega I)(\mathcal{A}^{\dagger}+i\omega I)\right)^{-1}$$

and $\mathbf{f}$ is the corresponding eigenvector.

Inverse power method with Laplacian preconditioning again:

$$
\begin{aligned}
f^{(k+1)} &= \left((\mathcal{A} - i\omega\mathcal{I})(\mathcal{A}^\dagger + i\omega\mathcal{I})\right)^{-1} f^{(k)} \\
(\mathcal{A} - i\omega\mathcal{I})(\mathcal{A}^\dagger + i\omega\mathcal{I}) f^{(k+1)} &= f^{(k)} \\
\mathcal{P}(\mathcal{A} - i\omega\mathcal{I})\mathcal{P}^{\dagger -1}\mathcal{P}^\dagger(\mathcal{A}^\dagger + i\omega\mathcal{I}) f^{(k+1)} &= \mathcal{P} f^{(k)}
\end{aligned}
$$

where $\mathcal{P}$ is the inverse of Laplacian or Stokes operator

Implement using actions with $\mathcal{P}$, $\mathcal{P}^\dagger$ and solves with $\mathcal{P}(\mathcal{A}-i\omega\mathcal{I})$, $\mathcal{P}^\dagger(\mathcal{A}^\dagger+ i\omega\mathcal{I})$:

$$
\begin{aligned}
g_1 &= \mathcal{P} f^{(k)} \\
\mathcal{P}(\mathcal{A} - i\omega\mathcal{I}) g_2 &= g_1 \\
g_3 &= \mathcal{P}^\dagger g_2 \\
\mathcal{P}^\dagger(\mathcal{A}^\dagger + i\omega\mathcal{I}) f^{(k+1)} &= g_3
\end{aligned}
$$

Tested on lid-driven cavity

Re=100

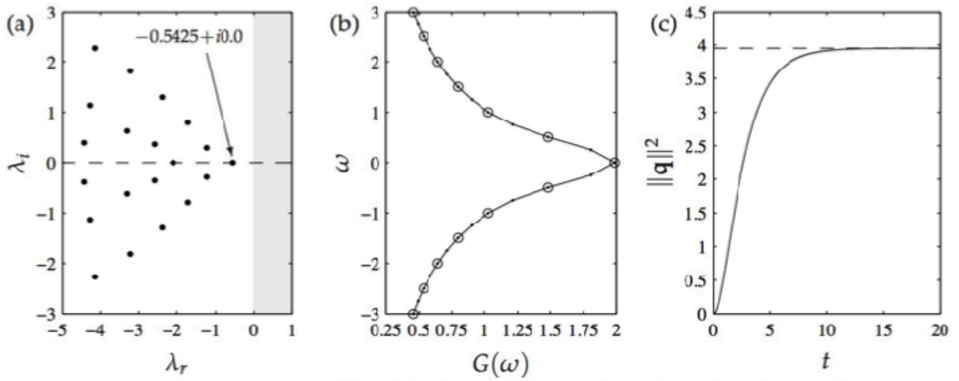


Figure 6: Frequency response of the lid-driven cavity at $Re = 100$. The eigenvalue spectra showing the 20 leading eigenvalues of the lid-driven cavity is plotted in frame (a) (the region of instability is colored in gray). The energy amplification for different frequencies is shown in frame (b), where results obtained with Algorithm 1 and 2 are plotted with dots and circles, respectively. Frame (c) shows the energy evolution in the system when driven by a steady force corresponding to the amplification peak in frame (b).

Re=8015

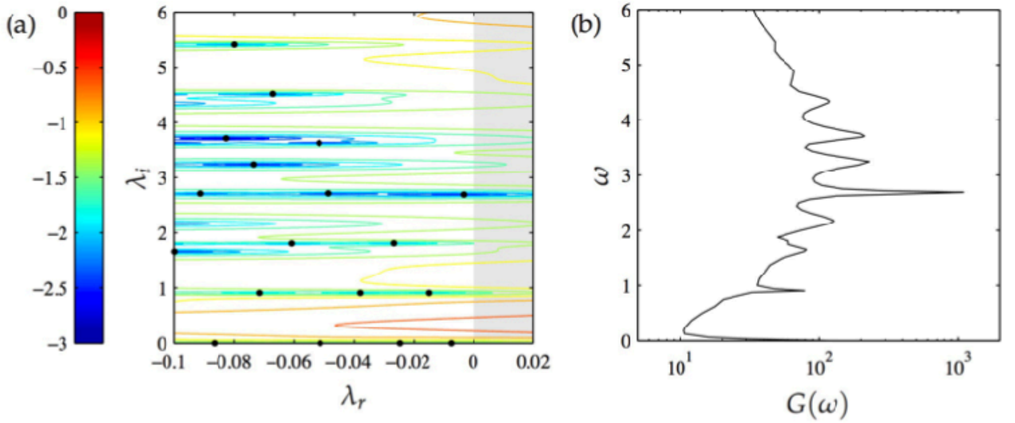


Figure 7: Frequency response of the lid-driven cavity at $Re = 8015$. The leading eigenvalues of the lid-driven cavity and the ε-pseudospectrum (logarithmically spaced coloured contours) are plotted in frame (a) (the region of instability is colored in gray). The energy amplification for different frequencies is shown in frame (b).

Forcing

Response

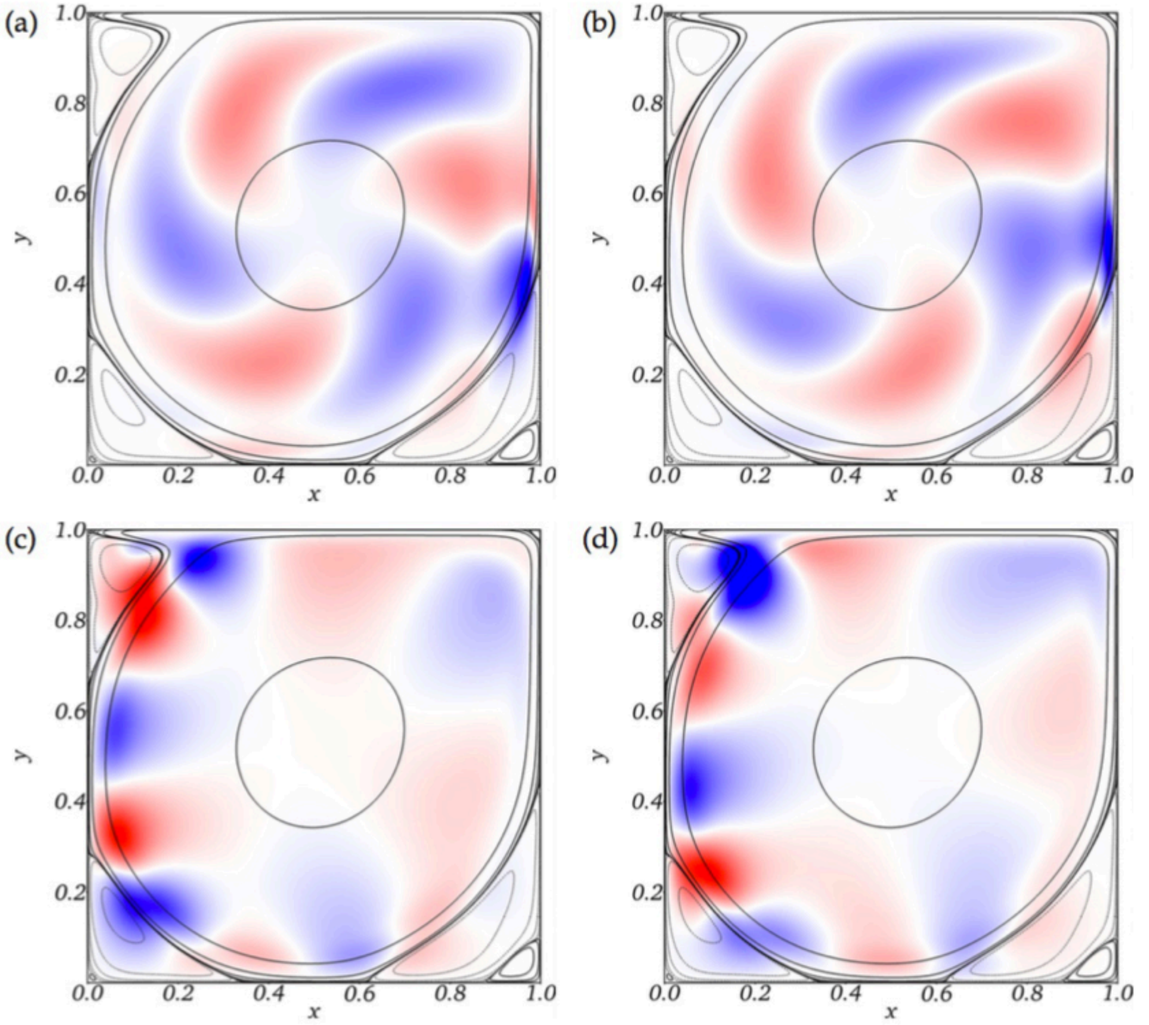


Figure 8: Optimal forcing profile at $\omega = 2.6875$ for the two-dimensional lid-driven cavity, $Re = 8015$. Real and imaginary parts of the optimal forcing profile are shown in (a) and (b), respectively, together with real and imaginary part of the resulting flow response shown in (c) and (d). The color shows the streamfunction of the optimal forcing and response, with red and blue indicating positive and negative values, respectively. The solid and dashed lines represent positive and negative values of logarithmically distributed contours of the baseflow streamfunction.

Performance of method for lid-driven cavity

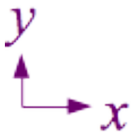
Competing method uses forwards and backwards time integration (e.g. Monokrousos, Akervik, Brandt, Henningson, JFM 2010)

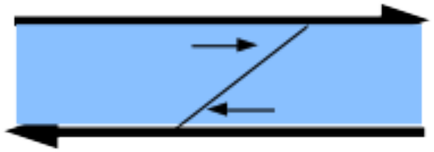
Table 1: Comparison of the results and the number of operator evaluations associated with Algorithm 1 and Algorithm 2 for different ω.

	Time integration			Inverse power method	
ω	T	cost	$G(\omega)$	cost	$G(\omega)$
0.0	29.00	261,000	1.987883	533	1.987877
1.0	31.42	471,240	1.029109	3,764	1.029304
3.0	14.66	425,140	0.454935	7,208	0.454855
5.0	7.54	452,400	0.275763	15,020	0.275634

But….

Wall-bounded shear flows

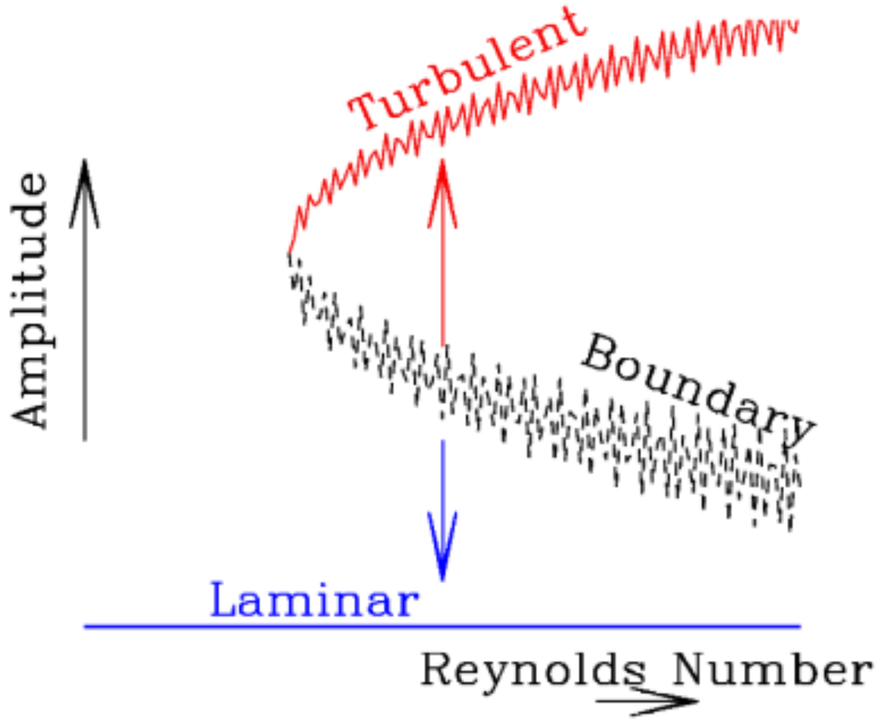


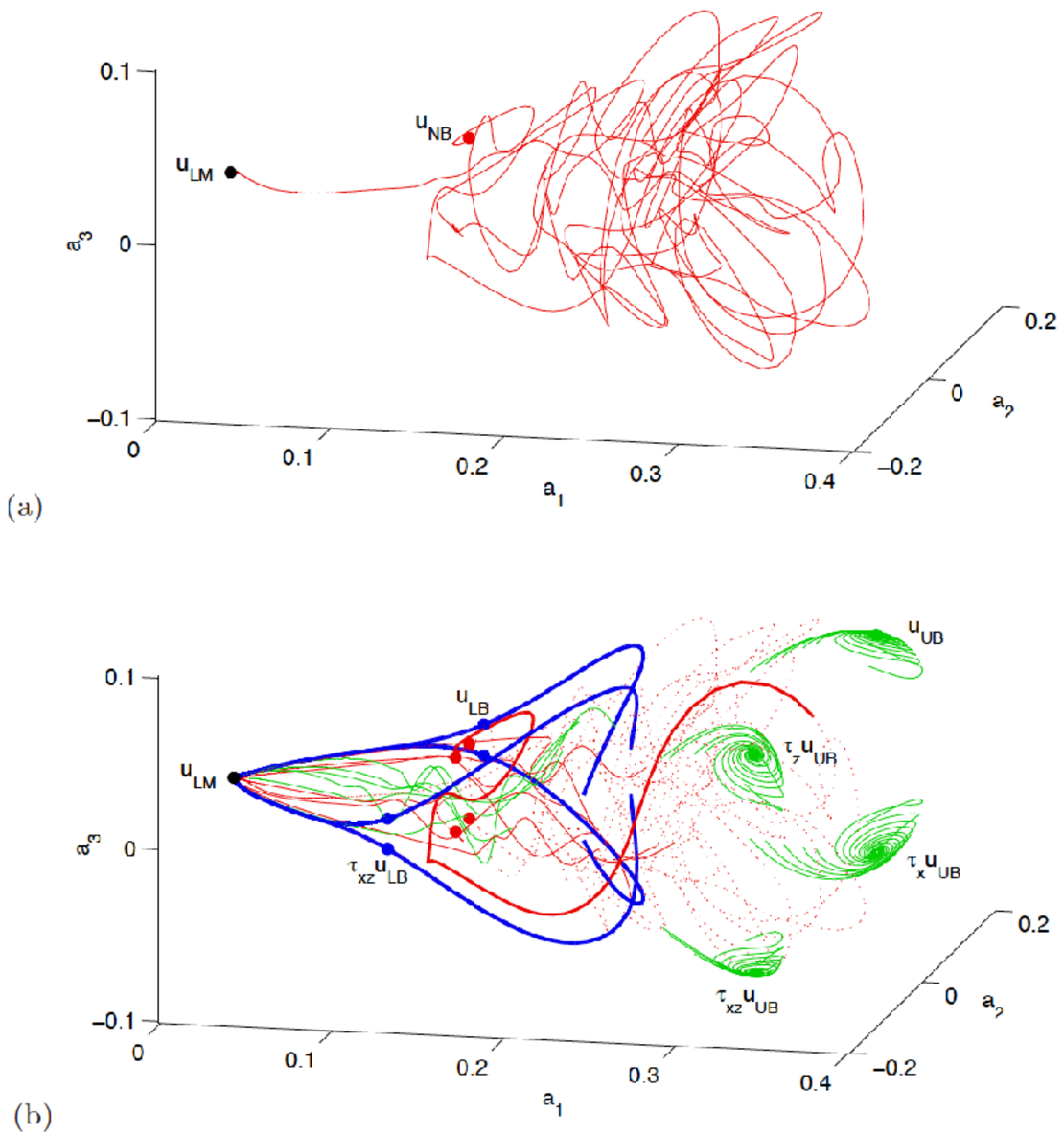






	Couette	**Pipe**	**Poiseuille**
linear instability:	Re = ∞ *(Romanov)*	Re = ∞	Re = 5772 *(Orszag)*
transition to turbulence:	Re ≈ 300	Re ≈ 2000	Re ≈ 1000
	$\boldsymbol{U}_{lam} = y\ \mathbf{e_x}$	$\boldsymbol{U}_{lam} = (1-r^2)\ \mathbf{e_z}$	$\boldsymbol{U}_{lam} = (1-y^2)\ \mathbf{e_x}$





For a long time, the only regular solutions known were the analytic laminar ones.

First new solution found numerically for plane Couette flow by Nagata (1990).

Then many many other solutions were found.

Is turbulence a chaotic attractor, with these solutions forming a skeleton?

Gibson, Halcrow, Cvitanovic, J. Fluid Mech. (2008)

Methods used in hydrodynamics for computing steady states and traveling waves

Newton with full direct Jacobian inversion:
from 1980s: Keller, Busse & Clever, Nagata, Generalis, Kerswell, Eckhardt
scales like M^3, so limited to small resolutions

Newton with matrix-free Jacobian inversion via Stokes preconditioning:
from 1989: Tuckerman, Henry/Bergeon/Beaume, Mercader, Mellibovsky

Newton with matrix-free Jacobian inversion via integration:
from 2004: Sanchez-Umbria, Viswanath, Duguet, Gibson, Willis

non-Newton or variants of Newton:
Newton-Picard with subspace iteration: Lust & Roose
Selective frequency damping: Akervik
Recursive projection method: Keller & Shroff

Navier-Stokes

$$\frac{\partial U}{\partial t} = LU + N(U)$$

Steady states

$$0 = LU + N(U)$$

Steady states OR T-periodic orbits

$$0 = U(T) - U(0) \equiv (\Phi_T - I)U(0)$$

In theory $\Phi_T : U(0) \to U(T)$

In practice $\Phi_T \approx \left(B_{\Delta t}\right)^{T/\Delta t}$

where a single small timestep is

$$B_{\Delta t} \equiv (I - \Delta t L)^{-1}(I + \Delta t N)$$

$$G_T^{\text{Stokes}} \equiv B_T - I$$

finds roots of

$$L + N = \frac{\partial U}{\partial t}$$

single timestep
per action of G

$$G_T^{\text{int}} \equiv \left(B_{\Delta t}\right)^{T/\Delta t} - I$$

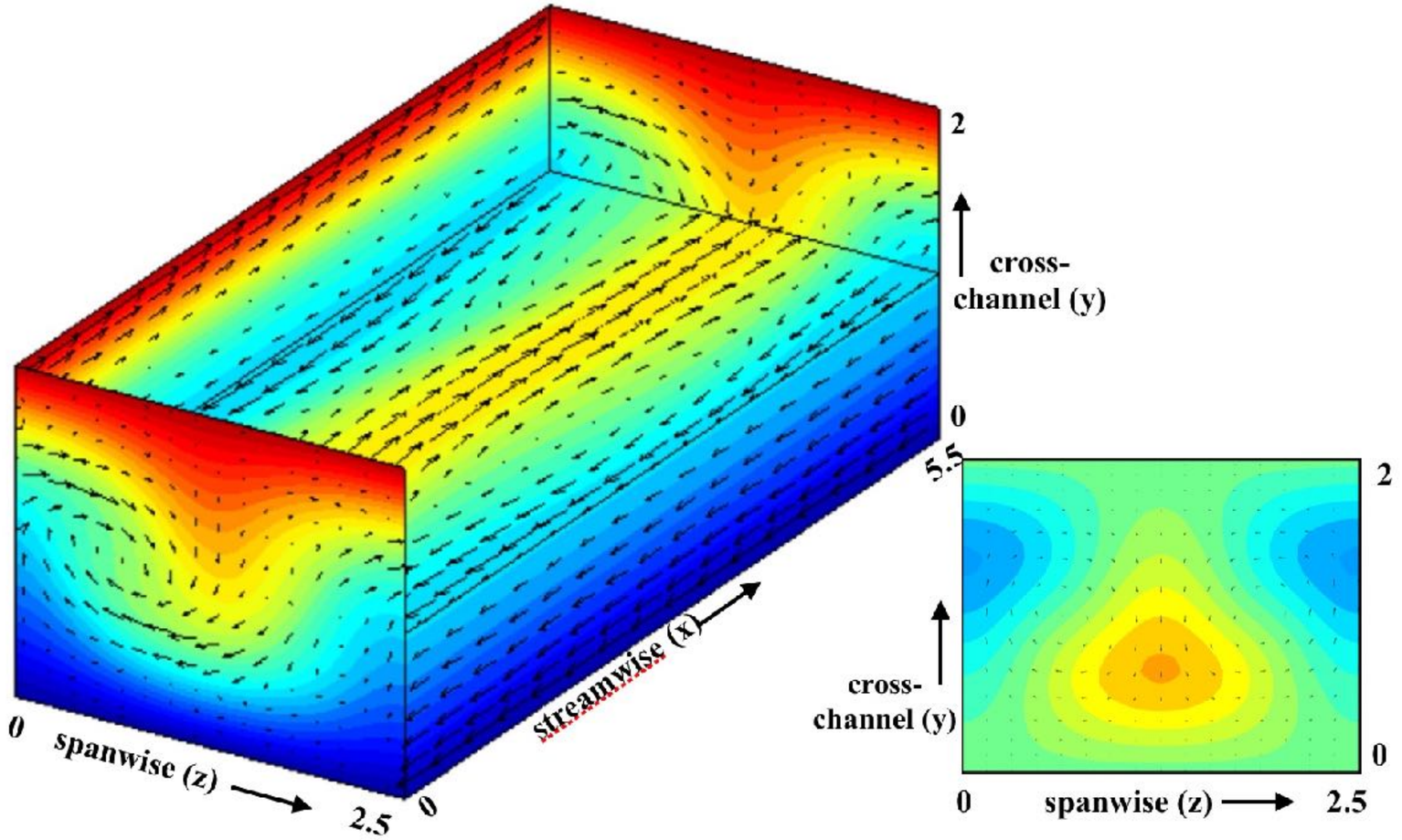
finds roots of

$$\Phi_T - I = U(T) - U(0)$$

many timesteps (T/Δt)
per action of G

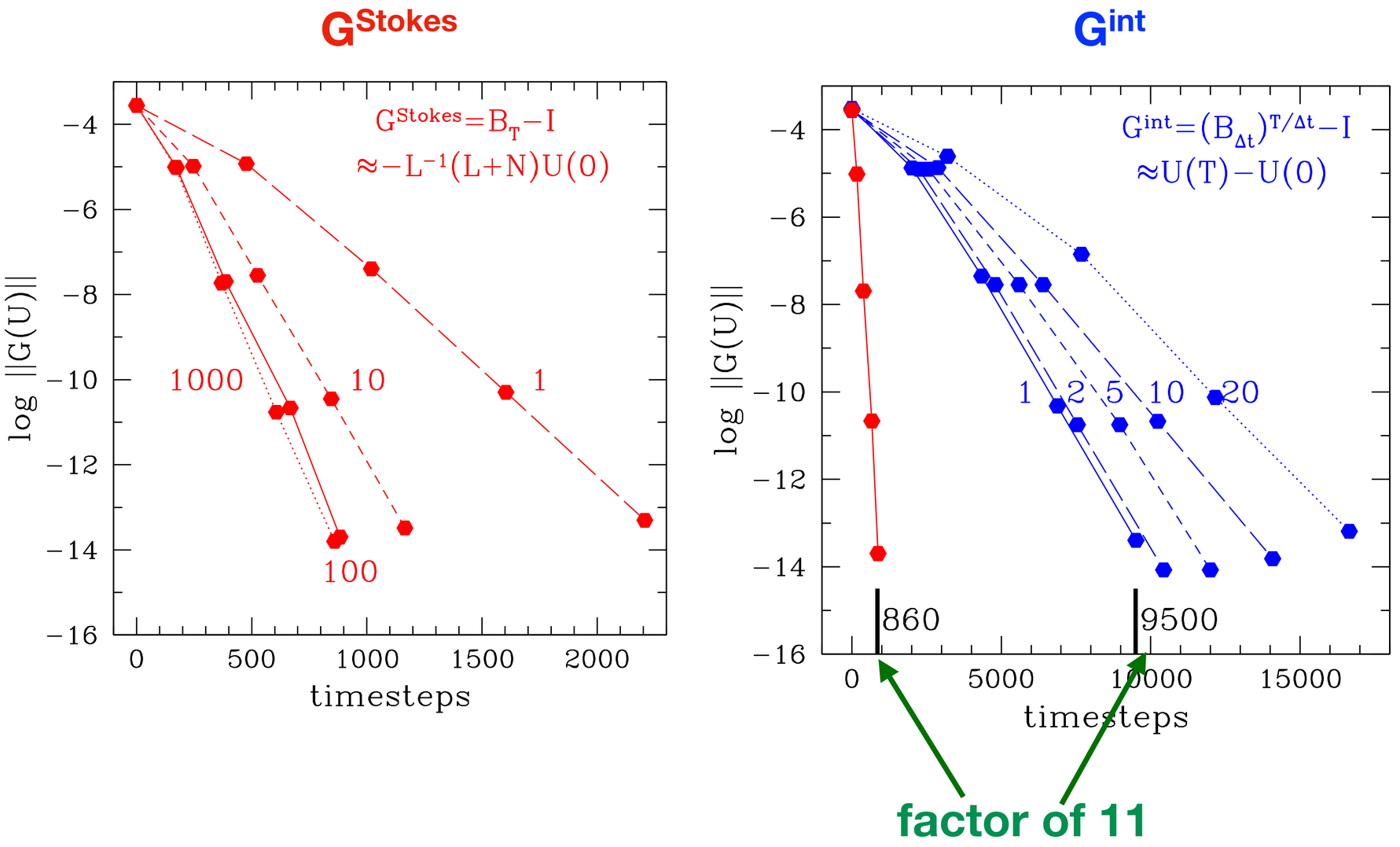
Steady state in plane Couette flow using Channelflow (Gibson)

Fourier x, Chebyshev y, Fourier z



Nagata solution (Gibson, Halcrow, Cvitanovic 2008)

Computation of steady state using Channelflow



Computation of traveling waves in pipe flow using Openpipeflow (Willis)
Fourier in θ, z / finite differences in r

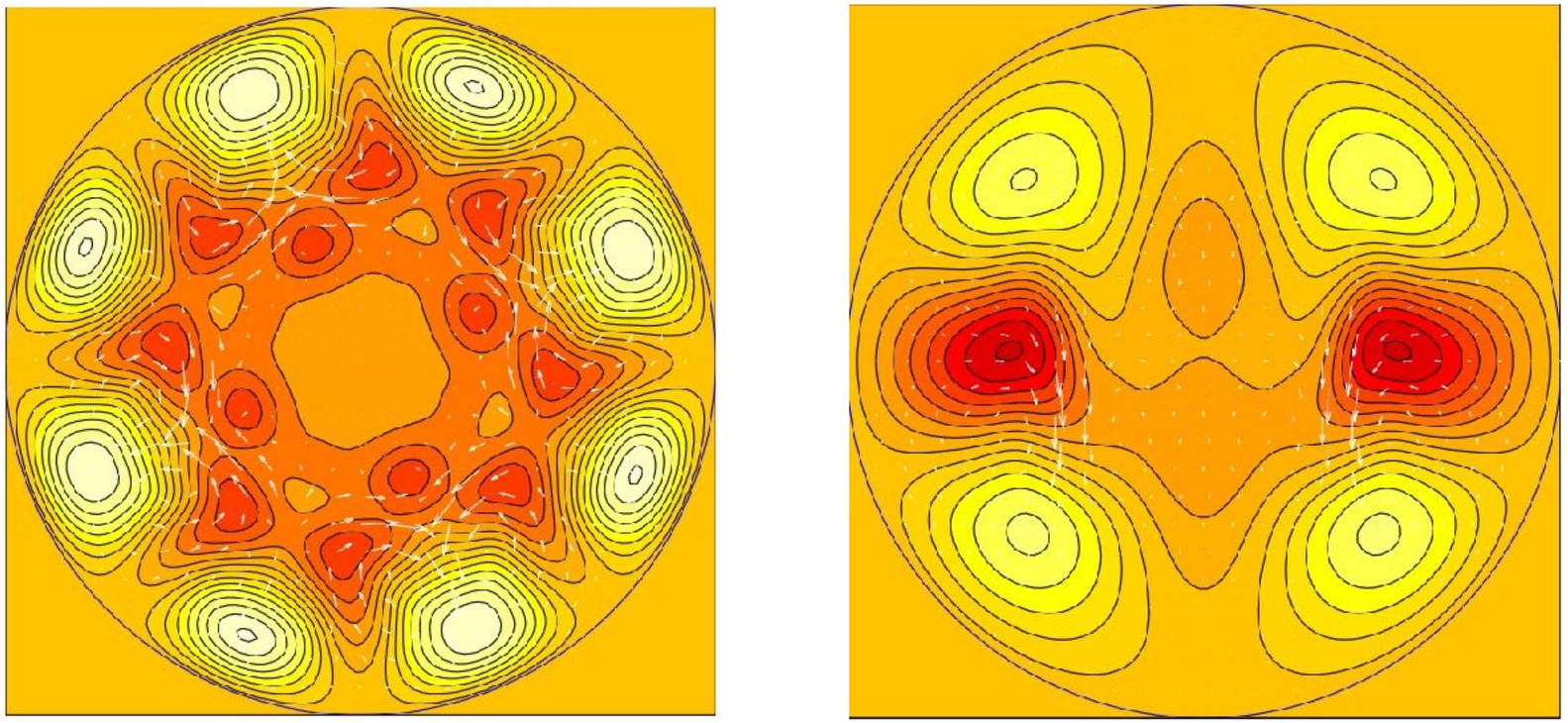
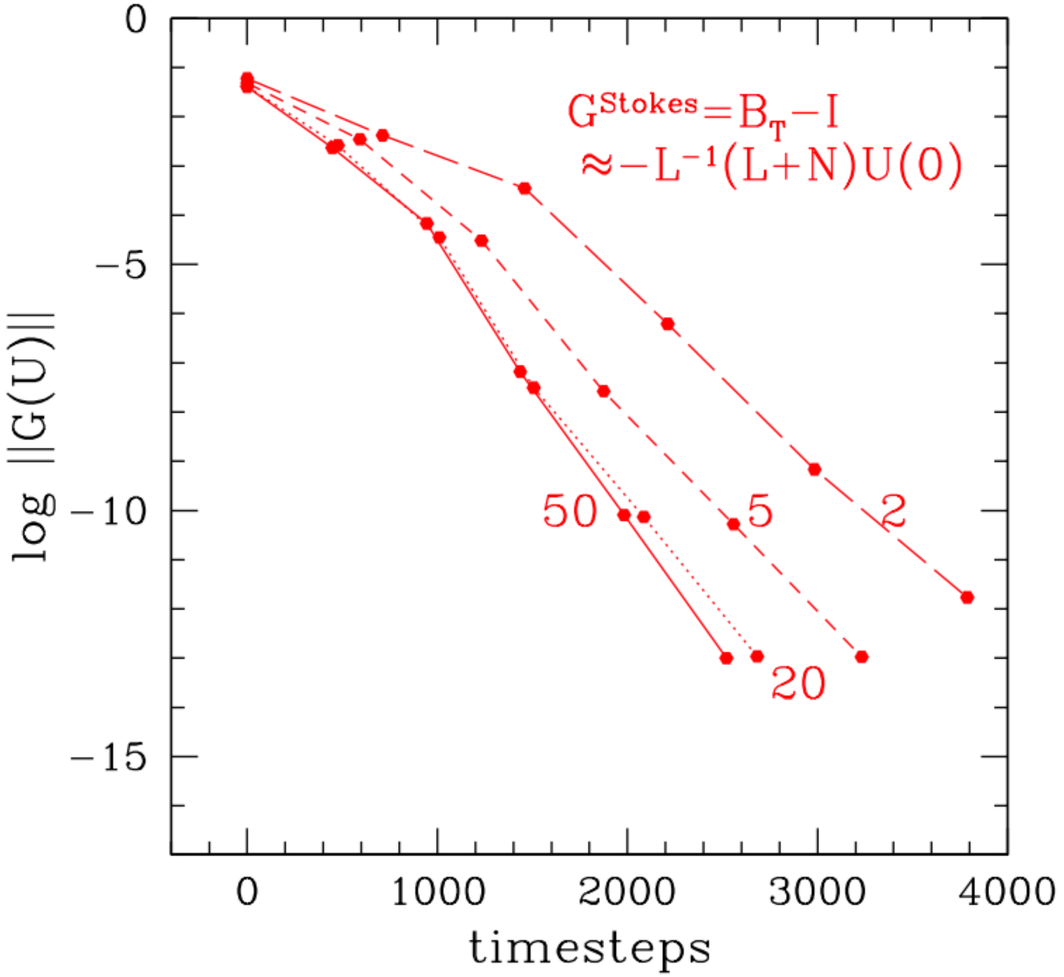
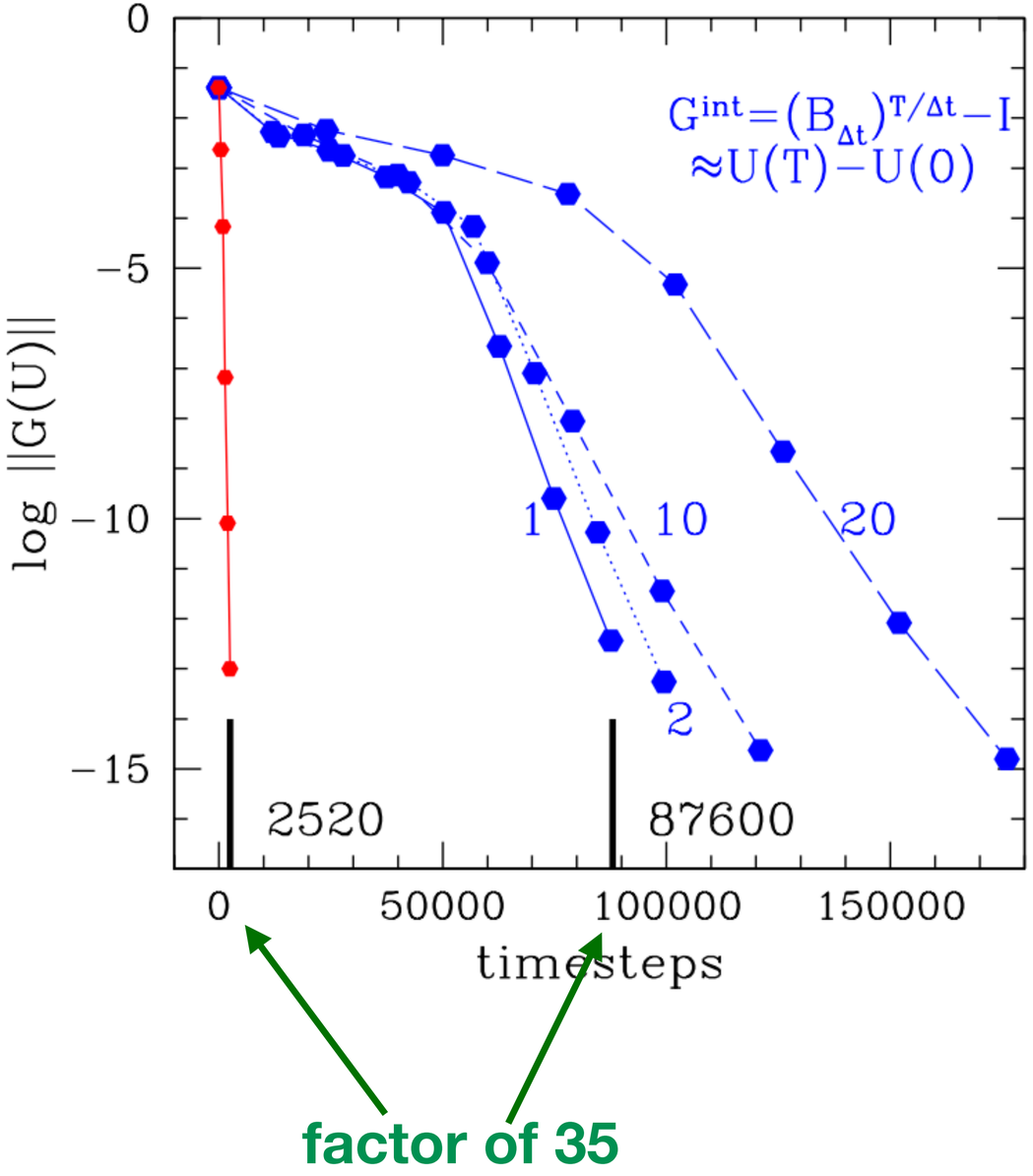


Fig. 4 Traveling wave states N4L (left) and M1L (right), visualized via the deviation of the streamwise velocity from the laminar profile. From Pringle, Duguet and Kerswell [87].

Computation of traveling waves using Openpipeflow



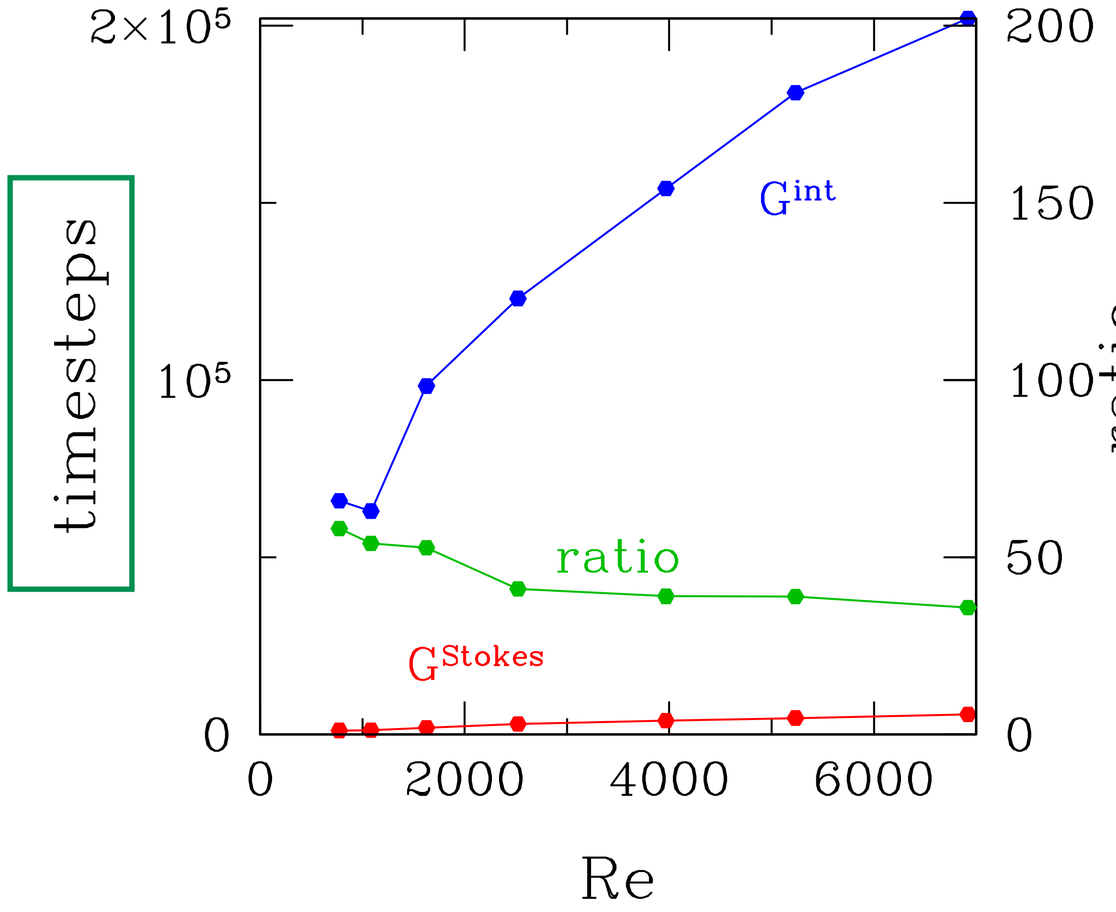


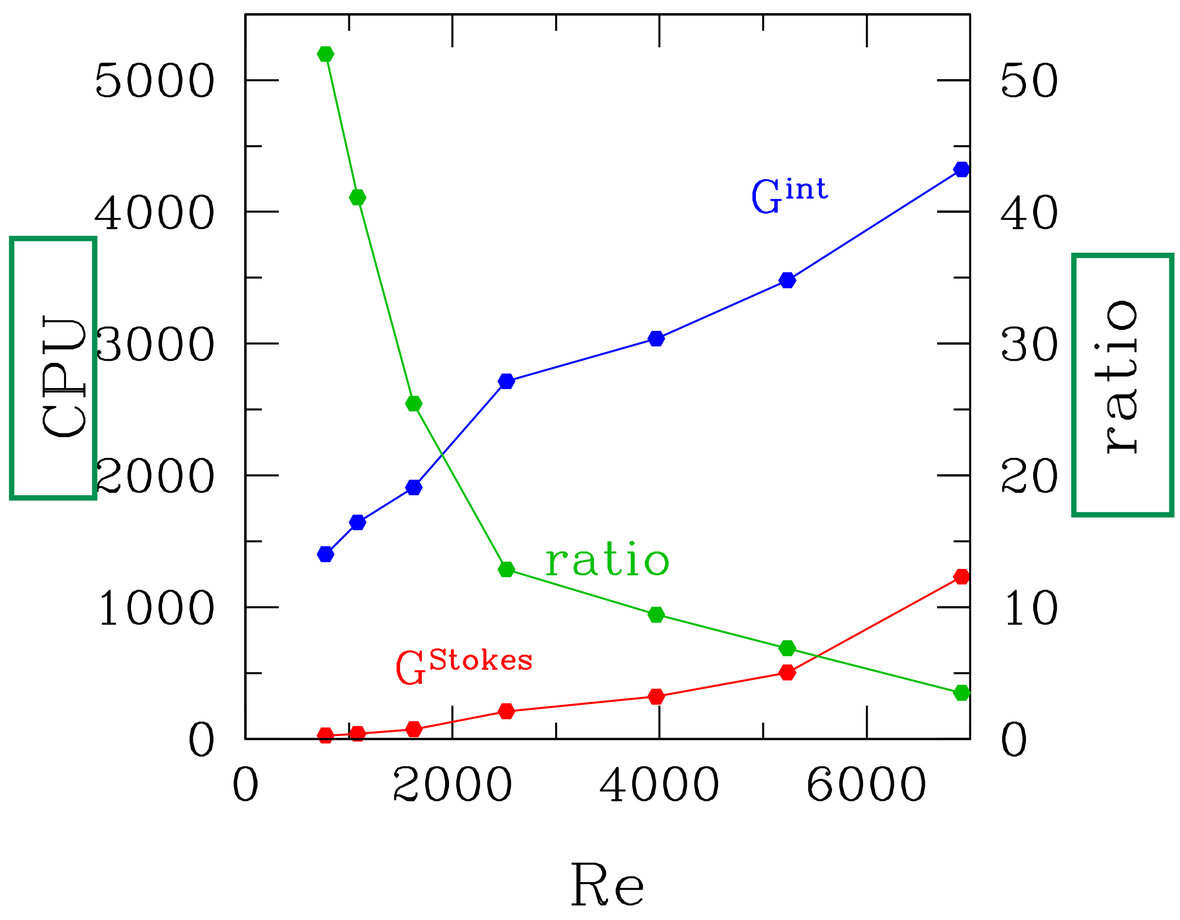
factor of 35

As Re increases,

the performance ratio measured by timesteps remains nearly constant.

But the performance ratio measured by CPU time decreases drastically.





Why?

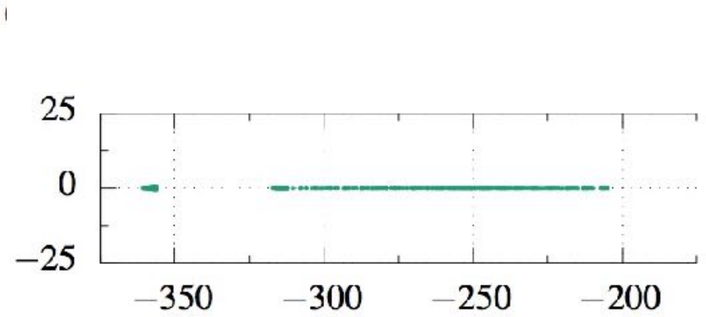
Operator spectra for Re = 500

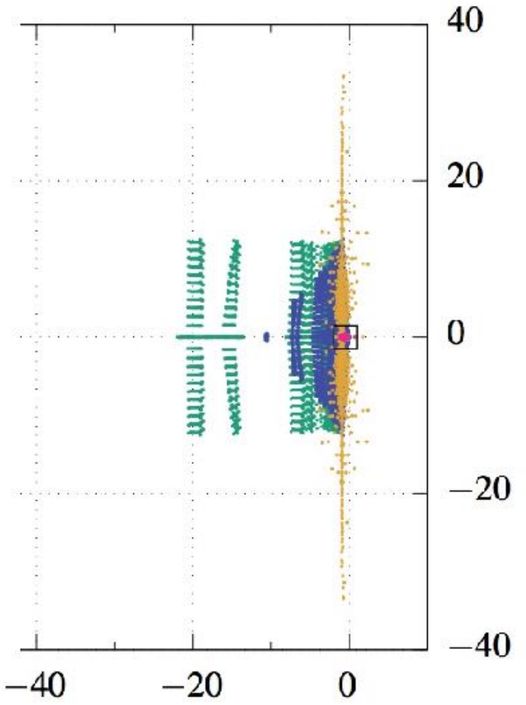
$G^{\text{Stokes}}_{0.0001}$

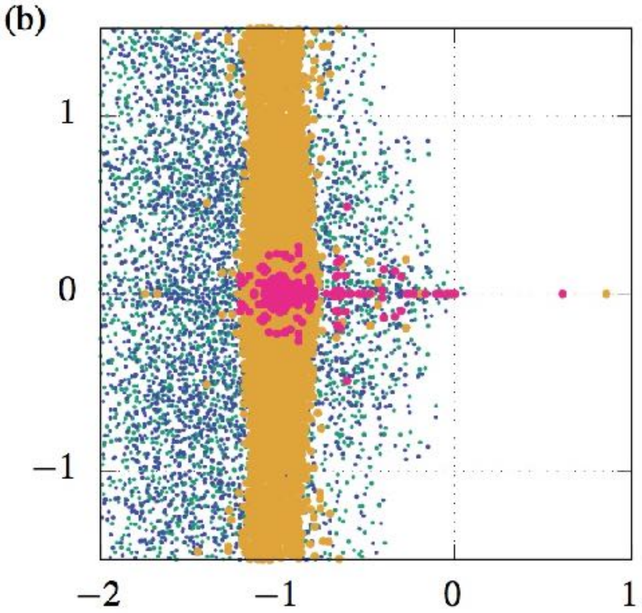
$G^{\text{Stokes}}_{0.1}$

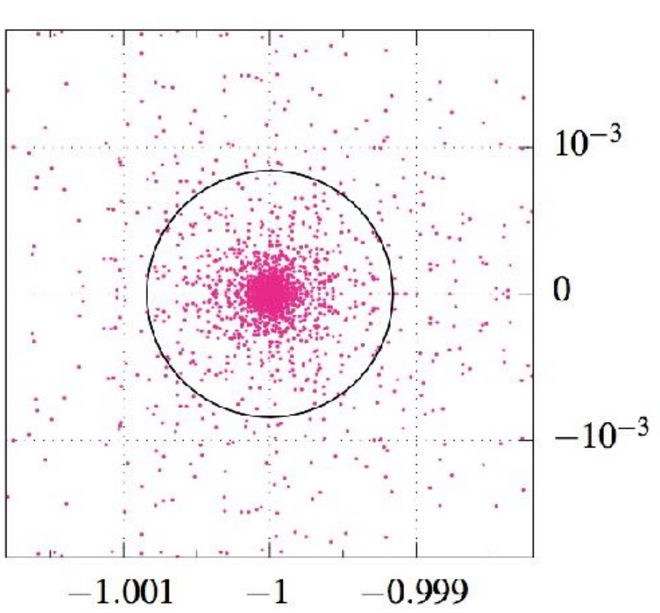
G^{Stokes}_{100}

G^{int}_{10}

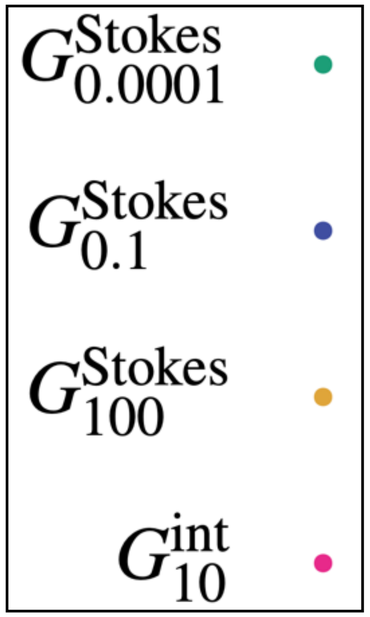


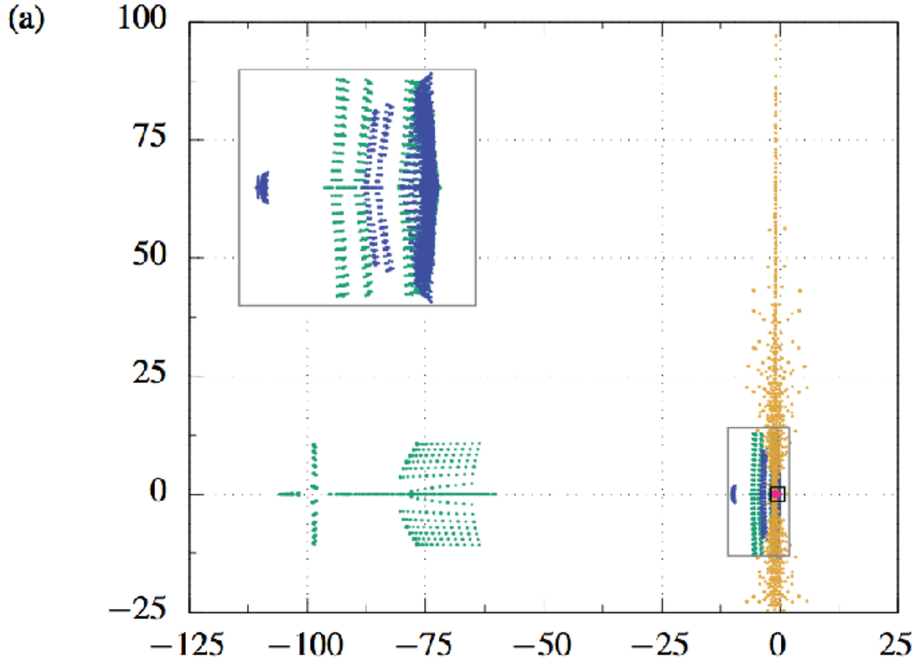


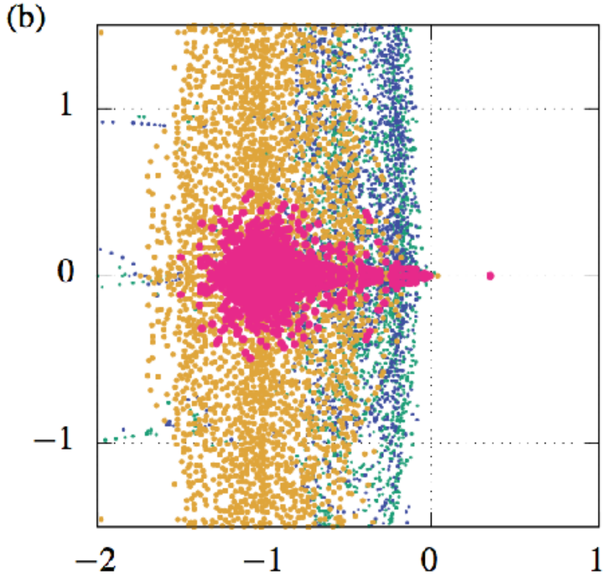


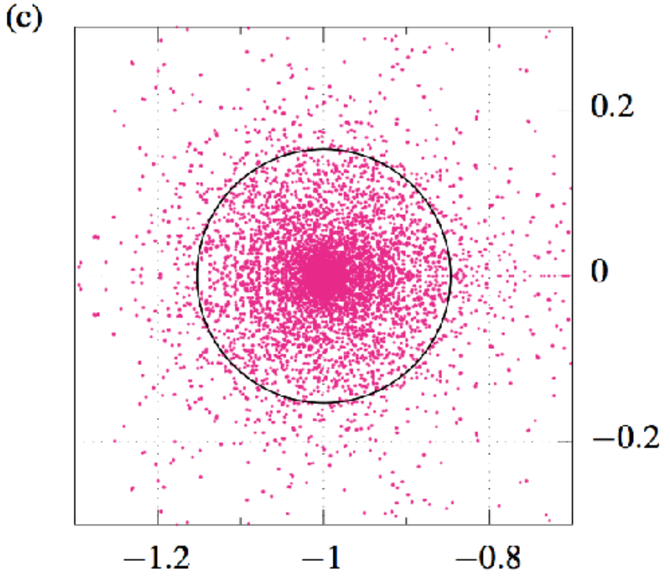


Operator spectra for Re = 1750









$$\left(\frac{1}{Re}\nabla^2 + U\partial_x\right) u$$

$$\lambda_k^{\text{model}} = -k^2/Re \pm ikU$$

$$\lambda^{\text{int}} \approx \exp(\lambda T) - 1 \approx \left\{ \begin{array}{c} \lambda T \\ -1 \end{array} \right\} \text{ for } \left\{ \begin{array}{l} |\lambda T| \ll 1 \\ |\lambda T| \gg 1, \text{ with } \lambda < 0 \end{array} \right\}$$

$$\lambda_k^{\text{Stokes}} \approx \frac{\lambda_k^{\text{model}}}{k^2/Re} = -1 \pm iURe/k$$

$$\lambda_k^{\text{Stokes}} \approx \frac{\lambda_k^{\text{model}}}{k^2/Re} = -1 \pm iURe/k$$

Seek to reduce large imaginary eigenvalues of G^{Stokes}

implicit $L\mathbf{u} = \frac{1}{Re}\nabla^2\mathbf{u}$ explicit $N_{\mathbf{U}}\mathbf{u} = -\left[\mathbf{U}\cdot\nabla)\mathbf{u} + (\mathbf{u}\cdot\nabla)\mathbf{U}\right]$

$$\mathbf{U} = U_{\text{lam}}(y)\,\mathbf{e}_x + \tilde{\mathbf{U}} \rightarrow N_{\mathbf{U}}\mathbf{u} = -\left[U_{\text{lam}}(y)\partial_x\mathbf{u} + vU'_{\text{lam}}(y)\hat{\mathbf{e}}_x + (\tilde{\mathbf{U}}\cdot\nabla)\mathbf{u} + (\mathbf{u}\cdot\nabla)\tilde{\mathbf{U}}\right]$$

term of N_U most responsible for bad conditioning AND easiest to make implicit !

$\longrightarrow$ incorporate $L\mathbf{u} \equiv \left(\frac{1}{Re}\nabla^2 - U_{\text{lam}}\partial_x\right)\mathbf{u}$

Another case: rapid rotation

$$\partial_t\mathbf{u} = -\nabla p - (\mathbf{u}\cdot\nabla)\mathbf{u} + \underbrace{\frac{1}{Re}\nabla^2\mathbf{u} - 2\boldsymbol{\Omega}\times\mathbf{u}}_{L\mathbf{u}}$$

SUMMARY

Time stepping	Steady-state solving	Linear stability analysis
$\partial_t U = (N+L)U$	$0 = (N+L)U$	$\lambda u = (N_U+L)u$
Implicit/explicit Euler	**Newton**	**Inverse power/Arnoldi**
$U(t+\Delta t) = BU(t)$ $= (I-\Delta t L)^{-1}$ $(I+\Delta t N)U(t)$	$(N_U+L)u$ $= (N+L)U$ $U \leftarrow U-u$	$(N_U+L)u_{n+1} = u_n$
$\equiv P(I+\Delta t N)U(t)$	$A_U u = AU$ $PA_U u = PAU$	$A_U u_{n+1} = u_n$ $PA_U u_{n+1} = Pu_n$
	$3-4$ **Newton steps**	$3-4$ **Inverse Arnoldi steps**
	200 BiCGSTAB iters/step	**200 BiCGSTAB iters/step**