Bifurcation Analysis for Time Steppers

Laurette Tuckerman

laurette@pmmh.espci.fr

THE THREE TOOLS OF COMPUTATIONAL FLUID DYNAMICS

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

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

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

Heat Equation

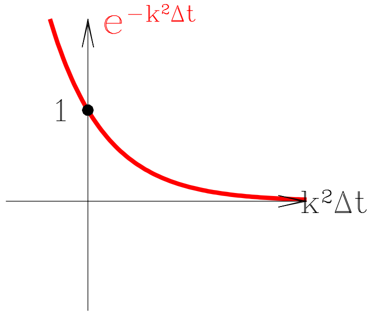
$$\partial_t u = \partial^2_{xx} u$$

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

$$\partial_t u_k = -k^2 u_k$$

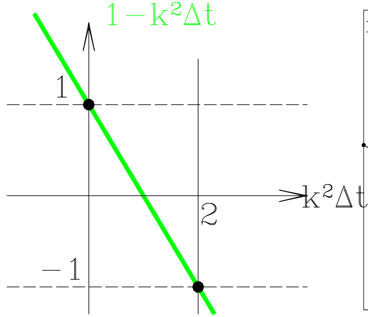
Exact solution

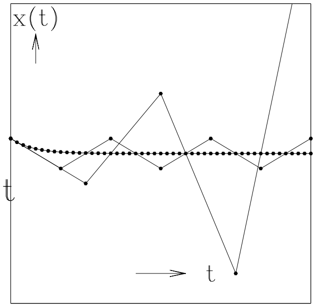
$$u_k(t+\Delta t) = e^{-k^2 \Delta t} u_k(t)$$



Explicit Euler

$$
\begin{aligned}
u_k(t+\Delta t) &= u_k(t) - k^2 \Delta t u_k(t) \\
&= (1 - k^2 \Delta t) u_k(t)
\end{aligned}
$$





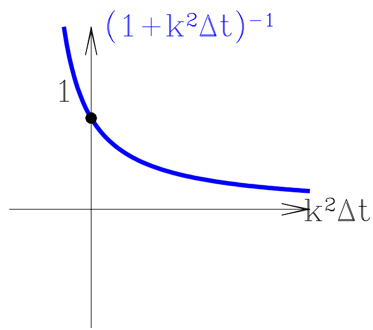
As $k_{max} \to \infty$, $\Delta t_{max} = \frac{2}{k_{max}^2} \to 0$

Implicit Euler

$$
\begin{aligned}
u_k(t+\Delta t) &= u_k(t) - k^2 \Delta t u_k(t+\Delta t) \\
(1 + k^2 \Delta t) u_k(t+\Delta t) &= u_k(t) \\
u_k(t+\Delta t) &= (1 + k^2 \Delta t)^{-1} u_k(t)
\end{aligned}
$$

Matrix version:

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



Navier-Stokes Equations

$$\begin{aligned}
\partial_t \boldsymbol{U} &= -(\boldsymbol{U}\cdot\boldsymbol{\nabla})\boldsymbol{U} - \boldsymbol{\nabla} \boldsymbol{P} &+& \boldsymbol{\nu}\boldsymbol{\nabla}^2\boldsymbol{U} \\
&= -(\boldsymbol{I} - \boldsymbol{\nabla}\boldsymbol{\nabla}^{-2}\boldsymbol{\nabla}\cdot)(\boldsymbol{U}\cdot\boldsymbol{\nabla})\boldsymbol{U} &+& \boldsymbol{\nu}\boldsymbol{\nabla}^2\boldsymbol{U} \\
&= \boldsymbol{N}(\boldsymbol{U}) &+& \boldsymbol{L}\,\boldsymbol{U}
\end{aligned}$$

- **Time stepping (Direct Numerical Simulation)**

$$\partial_t \boldsymbol{U} = \boldsymbol{N}(\boldsymbol{U}) + \boldsymbol{L}\,\boldsymbol{U} \equiv \boldsymbol{A}(\boldsymbol{U})$$

Explicit/Implicit Euler:

$$\begin{aligned}
\boldsymbol{U}(t+\Delta t) &= \boldsymbol{U}(t) + \Delta t[\boldsymbol{N}(\boldsymbol{U}(t)) + \boldsymbol{L}\boldsymbol{U}(t+\Delta t)] \\
&= (\boldsymbol{I} - \Delta t \boldsymbol{L})^{-1}(\boldsymbol{I} + \Delta t \boldsymbol{N})\boldsymbol{U}(t) \equiv \boldsymbol{B}\boldsymbol{U}(t)
\end{aligned}$$

- **Steady state solving**

$$0 = \boldsymbol{N}(\boldsymbol{U}) + \boldsymbol{L}\,\boldsymbol{U}$$

Newton's method:

$$\begin{cases} \boldsymbol{A_U u} = \boldsymbol{A}(\boldsymbol{U}) \\ \boldsymbol{U} \leftarrow \boldsymbol{U} - \boldsymbol{u} \end{cases}$$

- **Linear stability analysis:**

$$\boldsymbol{\lambda u} \;=\; \boldsymbol{N_U}\;\boldsymbol{u} + \boldsymbol{L}\;\boldsymbol{u} \equiv \boldsymbol{A_U u}$$

Arnoldi/block power method:

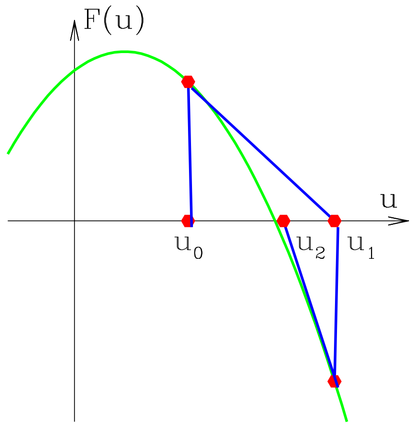
$$\boldsymbol{u}_{n+1} \;=\; \boldsymbol{A}_{\boldsymbol{U}}^{-1}\boldsymbol{u}_n \quad \textbf{or} \quad \boldsymbol{u}_{n+1} = e^{\boldsymbol{A_U}\Delta t}\boldsymbol{u}_n$$

$$\begin{aligned} \boldsymbol{N_U u} \;&\equiv\; -(\boldsymbol{U}\cdot\nabla)\boldsymbol{u} - (\boldsymbol{u}\cdot\nabla)\boldsymbol{U} \\ \boldsymbol{A_U u} \;&=\; \boldsymbol{N_U u} + \boldsymbol{L u} \end{aligned}$$

Steady state solving

$$0 = F(U)$$

Newton's method



$$0 = F(U - u) \approx F(U) - DF(U)u$$

$$\begin{cases} DF(U)u = F(U) \\ U \leftarrow U - u \end{cases}$$

Need to solve:

$$(N_U + L)u = s$$

where u could be a 3D field of size

$$3 \times 100 \times 100 \times 100 = 3 \times 10^6$$

Q: When are matrix operations cheap?

A: When the matrix is structured, e.g. tensor product.

Example: Fourier Transform

$$[\mathcal{F}_x\,\mathcal{F}_y\,\mathcal{F}_z]\,U(x,y,z) = \hat{U}(k_x,k_y,k_z)$$

Operation count:

$$M_x\,M_y\,M_z\,(M_x + M_y + M_z) \ll (M_x\,M_y\,M_z)^2$$

Actually even less:

$$M_x\,M_y\,M_z\,(\log M_x + \log M_y + \log M_z)$$

Poisson (or Helmholtz) equation:

$$\begin{aligned} s(x,y) &= \left(\partial_x^2 + \partial_y^2\right) p(x,y) \\ &= \left(E_x \Lambda_x^2 E_x^{-1} + E_y \Lambda_y^2 E_y^{-1}\right) p(x,y) \\ &= (E_x E_y) \left(\Lambda_x^2 + \Lambda_y^2\right) (E_x E_y)^{-1} p(x,y) \end{aligned}$$

because E_x, E_y commute.

Operation count:

$$M_x\, M_y\, (M_x + M_y) \rightarrow M_x\, M_y\, M_z\, (M_x + M_y + M_z)$$

Exponential (integration of heat equation):

$$\begin{aligned} U(x,y,t+\Delta t) = &\quad \exp\left(\Delta t(\partial_x^2 + \partial_y^2)\right) U(x,y,t) \\ = (E_x E_y) &\quad \exp\left(\Delta t(\Lambda_x^2 + \Lambda_y^2)\right) (E_x E_y)^{-1} U(x,y,t) \end{aligned}$$

Acting with ∇^2 or ∇^{-2} or $\exp(\Delta t \nabla^2)$ is inexpensive in a tensor-product grid for spectral or finite-difference methods.

Finite element and spectral element methods have their own tricks for fast direct action/inversion.
(Static condensation? Schur complement?)

Applies to $\nabla^2 = L$, not full Jacobian.

Used for implicit time-integration of diffusive/viscous step.

How to solve linear systems?

1) Direct: Gaussian Elimination = LU + Backsolve

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: Conjugate Gradient methods

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

For an arbitrary matrix,

$$\left.\begin{array}{l}\text{Each product } u \to Au \text{ requires } M^2 \text{ operations}\\ \text{Convergence requires } M \text{ iterations}\end{array}\right\} M^3$$

Can gain:

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

If A **is well-conditioned, convergence takes few iterations.**

Conditioning

A is well conditioned if its eigenvalues lie close together.

The best conditioned matrix is a multiple of the identity.

The condition number is, roughly,

$$\kappa(A) \sim \left| \frac{\text{max eig of } A}{\text{min eig of } A} \right|$$

Preconditioning

$$Au = v$$
$$PAu = Pv$$

where:

P is easy to act with

PA is better conditioned than A

Extreme cases:

$P = I$ (easy to act with but no improvement)

$P = A^{-1}$ (perfect preconditioner but impossible)

Our case:

$$A_U u = L u + N_U u$$

For 3D case with $M_x = M_y = M_z = 100$,
eigs of L range from ~ -1 to $-(M_x^2 + M_y^2 + M_z^2) = -30000$.

Idea:

L is the main source of difficulty $\Longrightarrow$ Use $P = L^{-1}$

Question: Where do we get L^{-1}?

Answer: Already present in a timestepping code!

$$\begin{aligned} U(t+\Delta t) - U(t) &= \left[(I-\Delta t L)^{-1}(I+\Delta t N) - I\right] U(t) \\ &= (I-\Delta t L)^{-1}\left[I + \Delta t N - (I - \Delta t L)\right] U(t) \\ &= (I-\Delta t L)^{-1} \Delta t (N+L)\, U(t) \end{aligned}$$

$(B-I)U(t) \equiv U(t+\Delta t) - U(t)$ **has same roots as** $(N+L)$**!**

- **In time-stepping,** Δt **must be small** $(\sim 10^{-2})$ **to insure** $(I-\Delta t L)^{-1}(1+\Delta t N_U) \approx \exp((L+N_U)\Delta t)$**.**
- **Here,** Δt **plays algebraic role, and can (should) be large** $(\gtrsim 10^2)$**.**
- Δt **interpolates between** $P = I$ **and** $P = L^{-1}$**.**
- **Called Stokes preconditioning**

One Newton Step

$$\color{red}{(\boldsymbol{I} - \Delta t \boldsymbol{L})^{-1}} \Delta t (\boldsymbol{N}_{\boldsymbol{U}} + \boldsymbol{L})\, \boldsymbol{u} \;=\; \color{red}{(\boldsymbol{I} - \Delta t \boldsymbol{L})^{-1}} \Delta t (\boldsymbol{N} + \boldsymbol{L})\, \boldsymbol{U}$$

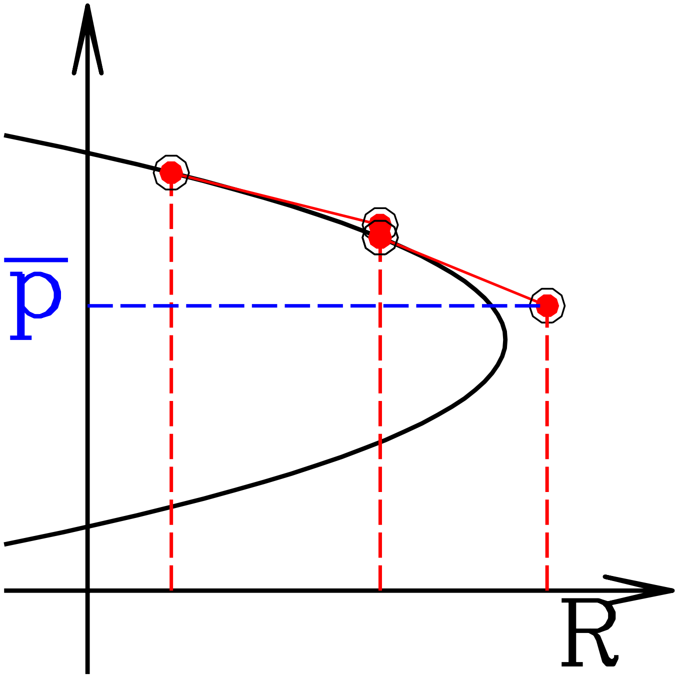
$$\underbrace{\left[\color{red}{(\boldsymbol{I} - \Delta t \boldsymbol{L})^{-1}} (\boldsymbol{I} + \Delta t \boldsymbol{N}_{\boldsymbol{U}}) - \boldsymbol{I}\right] \boldsymbol{u}}_{\text{difference between two widely spaced consecutive linearized timesteps}} \;=\; \underbrace{\left[\color{red}{(\boldsymbol{I} - \Delta t \boldsymbol{L})^{-1}} (\boldsymbol{I} + \Delta t \boldsymbol{N}) - \boldsymbol{I}\right] \boldsymbol{U}}_{\text{difference between two widely spaced consecutive timesteps}}$$

Solve linear system with Bi-CGSTAB

H.A. van der Vorst, *Bi-CGSTAB: A fast and smoothly converging variant of Bi-CG for the solution of nonsymmetric linear systems*, **SIAM J. Sci. Stat. Comput. 13, 631 (1992)**

$\sim$ **30 lines of code**

Continuation



Goal:

$$
\begin{aligned}
0 &= RN(U) + LU \\
0 &= p(U, R) - \bar{p} \quad \text{where} \left\{ \begin{array}{l} U_i \text{ some component} \\ R \end{array} \right\}
\end{aligned}
$$

Newton step:

(U, R) not solution, so try $(U - u, R - r)$

$$
\begin{aligned}
0 &= (R - r)N(U - u) + L(U - u) \\
&= RN(U) + LU - RN_U u - rN(U) - Lu + O(r, u)^2 \\
0 &= p(U - u, R - r) - \bar{p} = \left\{ \begin{array}{c} U_i - \bar{p} - u_i \\ R - \bar{p} - r \end{array} \right\}
\end{aligned}
$$

$$\underbrace{\left[\begin{array}{c|c} RN_U + L & N(U) \\ \hline 0\ 0 \ldots 0\ 1\ 0 \ldots 0 & 1 \end{array}\right]}_{or} \left[\begin{array}{c} u \\ r \end{array}\right] = \left[\begin{array}{c} RN(U) + LU \\ \left\{\begin{array}{c} U_i - \bar{p} \\ R - \bar{p} \end{array}\right\} \end{array}\right]$$

If $p(U, R) = R$ (i.e. set Reynolds number), then set $R = \bar{p}$, $r = 0$ and get previous case:

$$L^{-1}\left[RN_U + L\right][u] = L^{-1}\left[RN(U) + LU\right]$$

If $p(U, R) = U_i$, then must solve extended system for (u, r).

$$\left[\begin{array}{c|c} L^{-1}(RN_u + L) & L^{-1}N(U) \\ \hline 0\ 0 \ldots 0\ 1\ 0 \ldots 0 & 0 \end{array}\right] \left[\begin{array}{c} u \\ r \end{array}\right] = \left[\begin{array}{c} L^{-1}(RN(U) + LU) \\ U_i - \bar{p} \end{array}\right]$$

Set $u_i = U_i - \bar{p}$
Calculate $L^{-1}(RN_U + L)u$
Add $L^{-1}N(U)r$

} **Use only vectors and operators of length M**

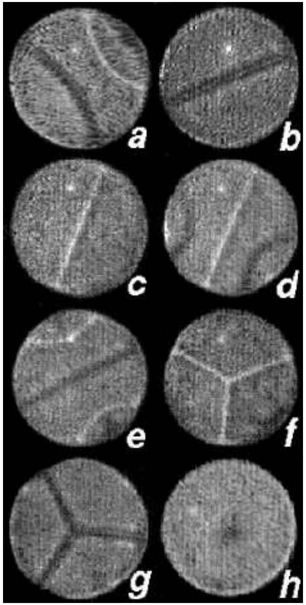
Traveling waves: $\boldsymbol{U(x - Ct, y, z)}$

$$\textbf{Goal}: \begin{cases} 0 = \boldsymbol{C\partial_x U + N(U) + LU} \\ 0 = \boldsymbol{p(U) - \bar{p}} \end{cases}$$

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

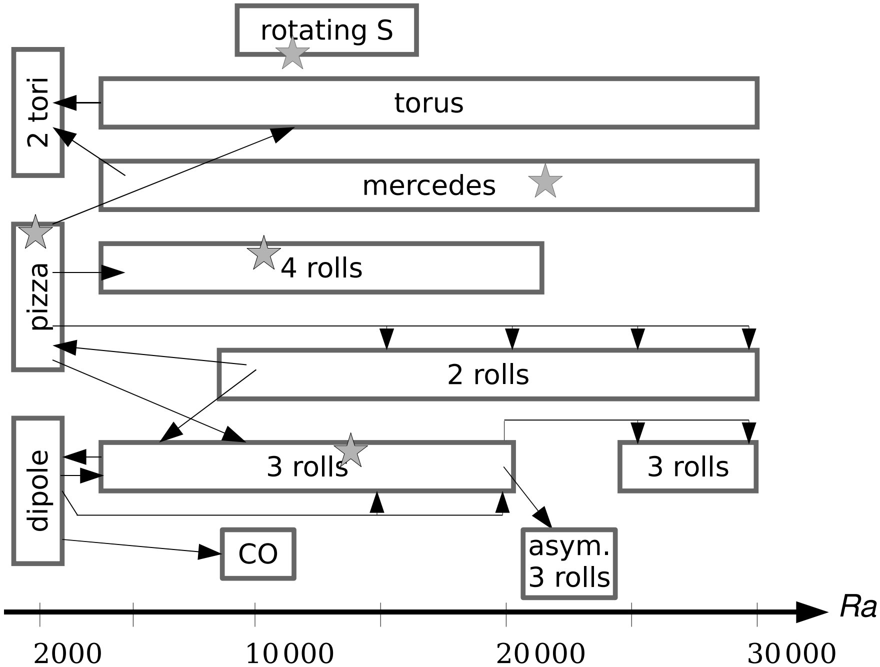
Extreme Multiplicity in Cylindrical Rayleigh-Bénard Convection

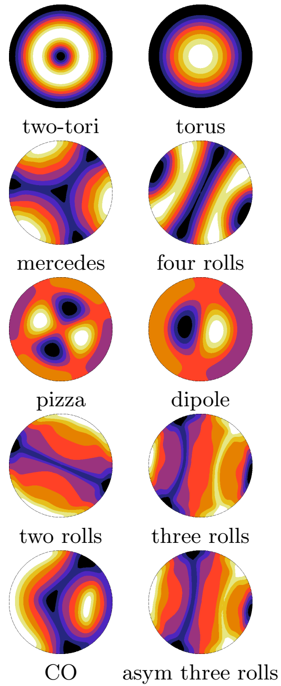
with K. Borońska



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

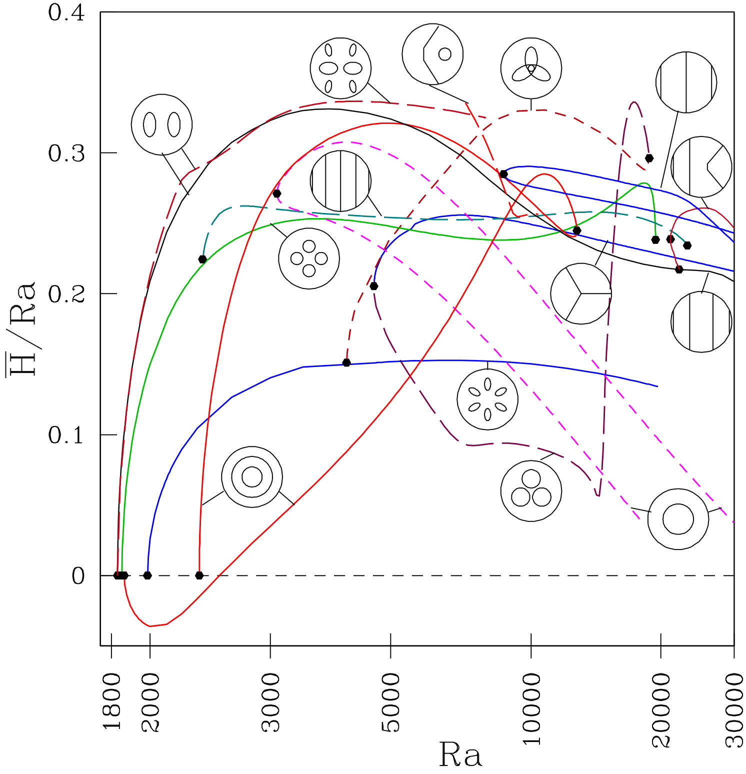
Results from Time-Dependent Simulations



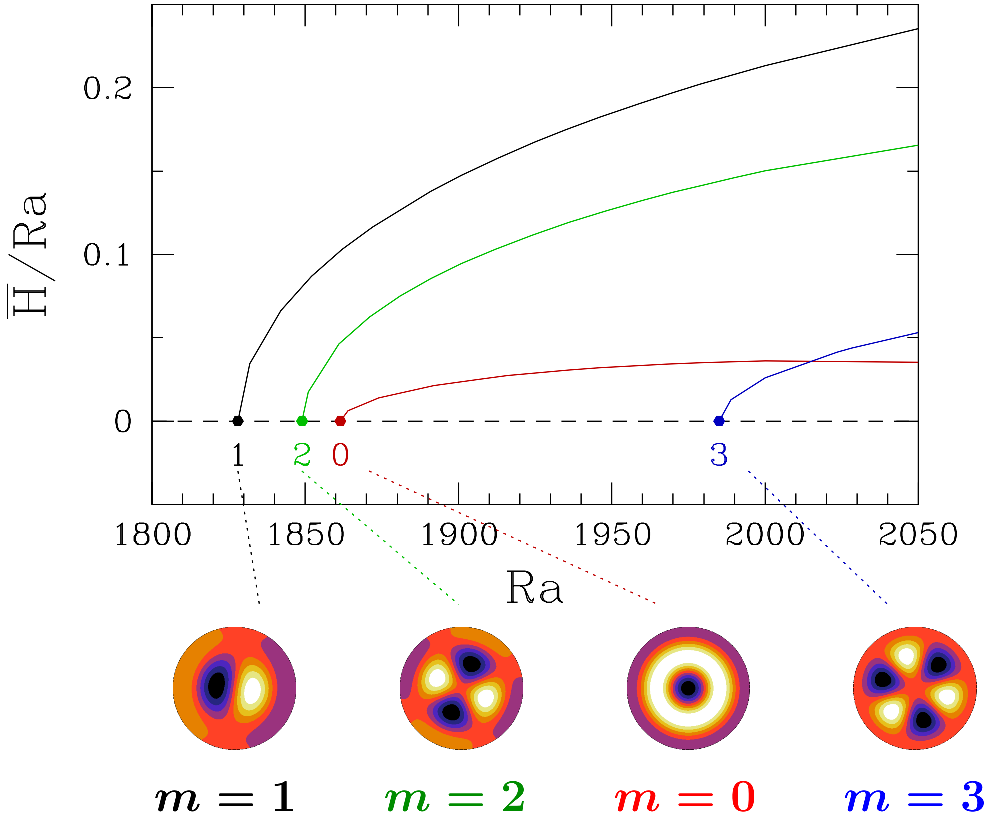


$$(U, V, W, T) = 4 \times N_r \times N_\theta \times N_z = 4 \times 40 \times 120 \times 20 = 384\,000$$

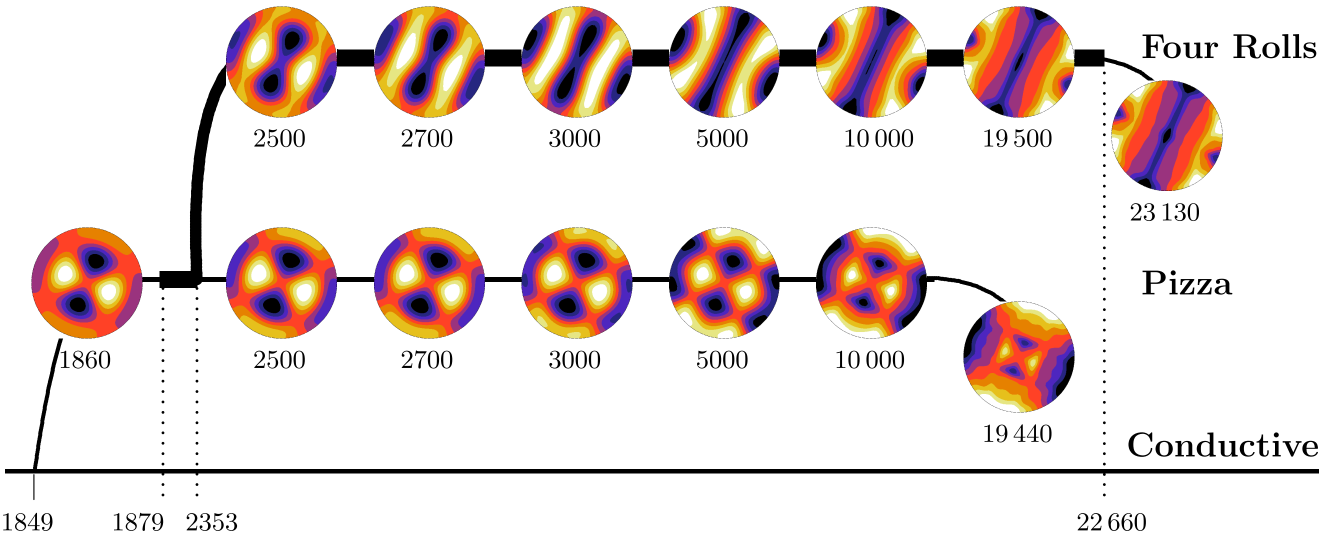
Complete Bifurcation Diagram



Thresholds from Conductive State

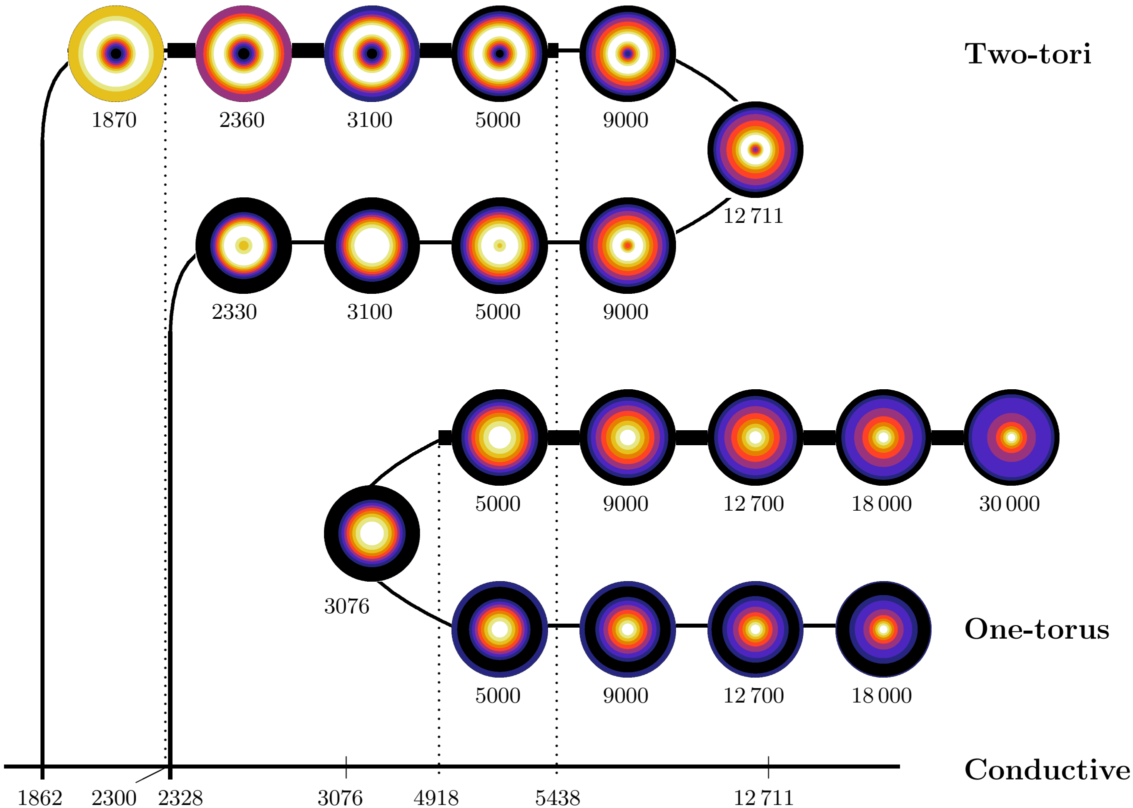


Pizza: Bifurcations from $m = 2$ mode

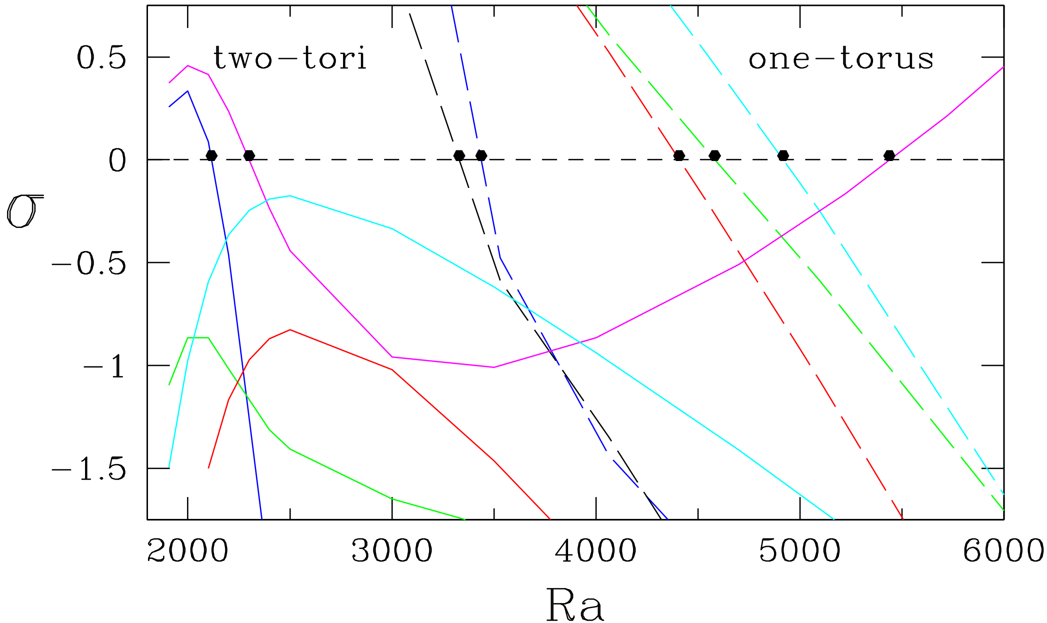


Trigonometric $\Longrightarrow$ *Rolls*

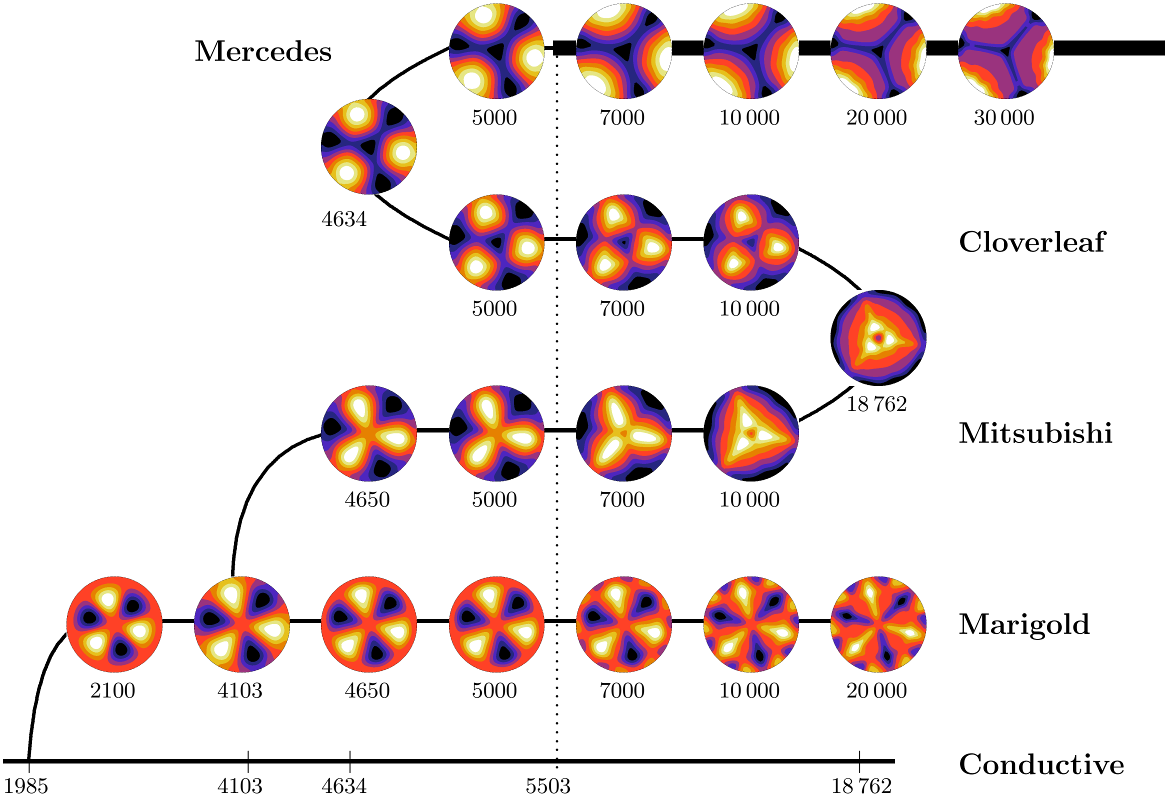
Tori: Bifurcations from $m = 0$ mode



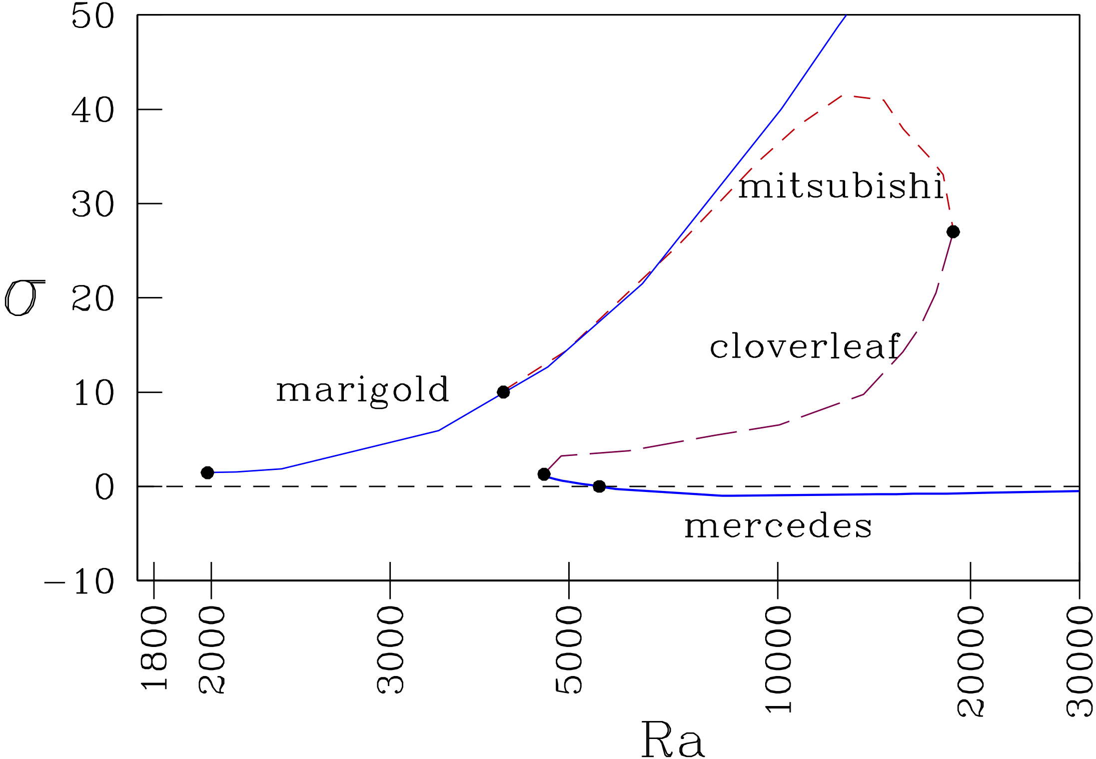
Stability of the $m = 0$ branches



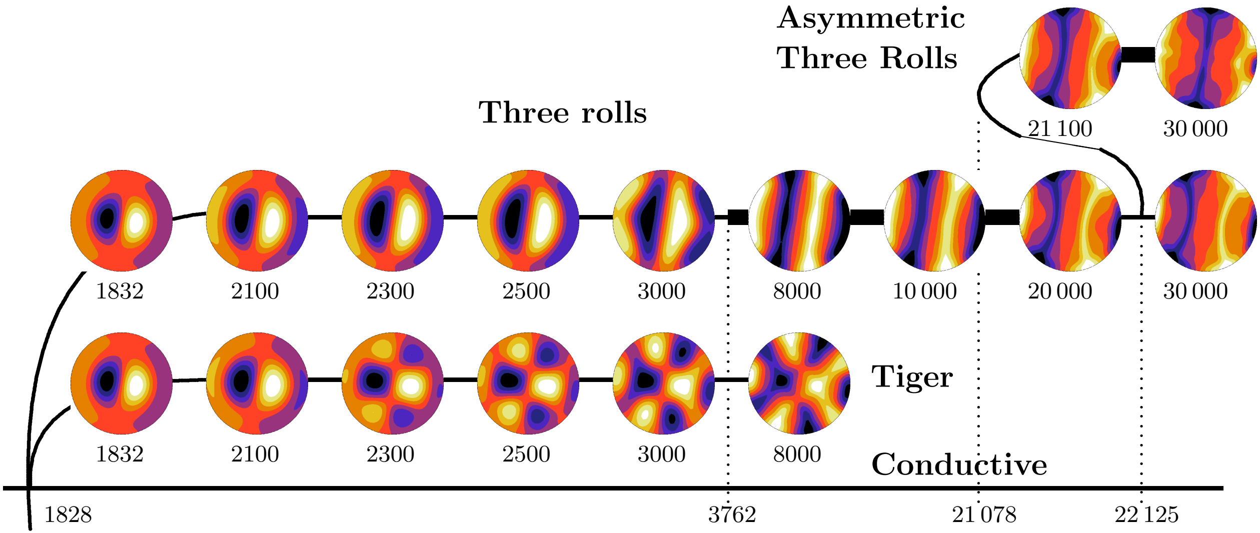
Flowers and Automobiles: Bifurcations from $m = 3$



Stability of the $m = 3$ *branches*

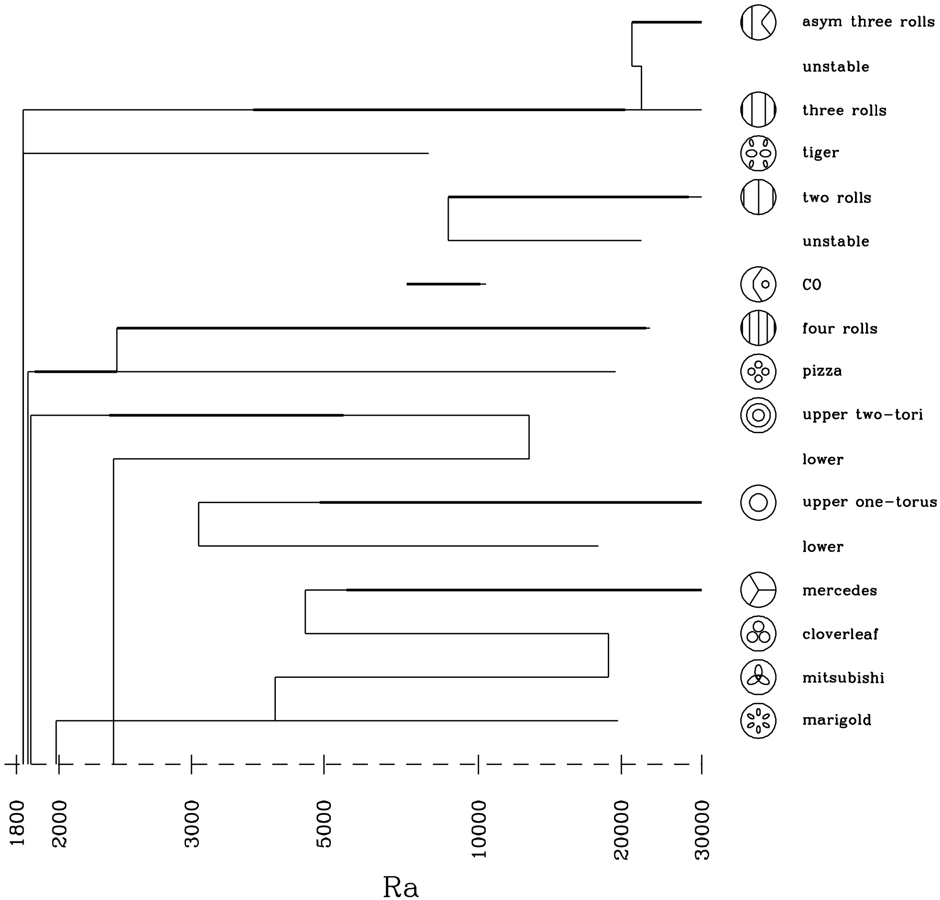


Dipoles: Bifurcations from $m = 1$ mode



Two branches with same symmetry bifurcate from

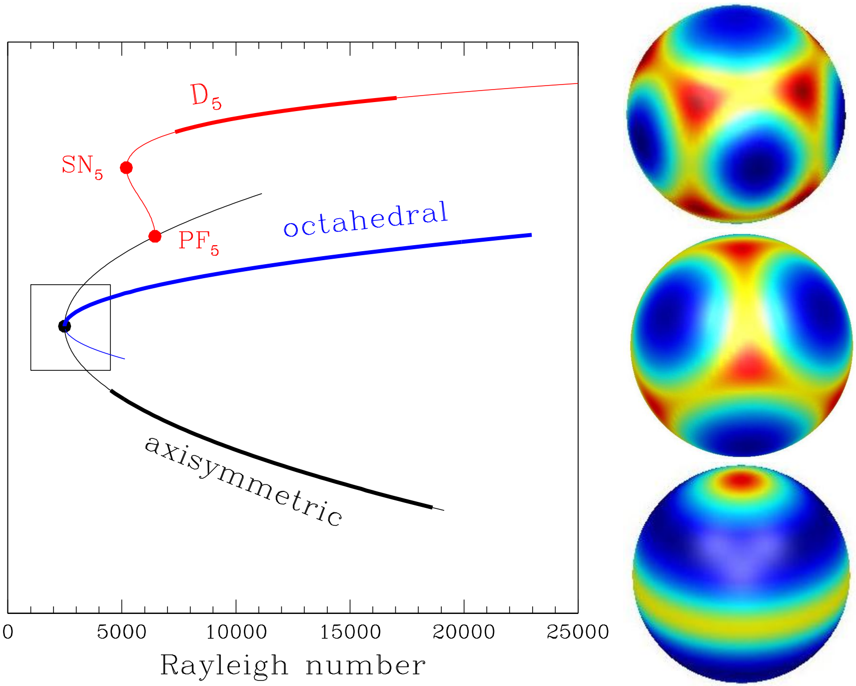
$$m \underset{?}{=} 1$$



asym three rolls
unstable
three rolls
tiger
two rolls
unstable
CO
four rolls
pizza
upper two-tori
lower
upper one-torus
lower
mercedes
cloverleaf
mitsubishi
marigold
1800
2000
3000
5000
10000
20000
30000
Ra

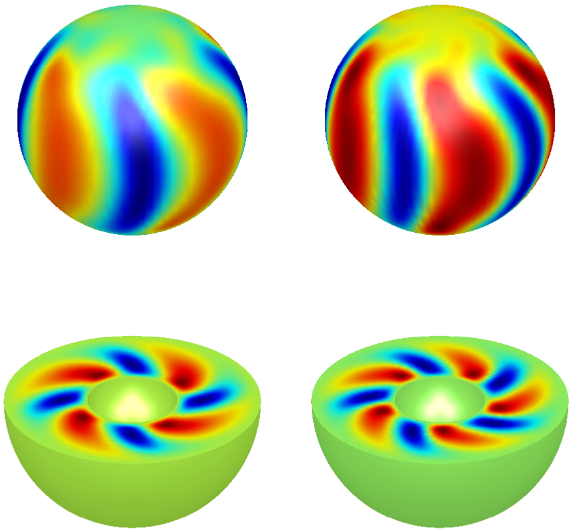
with Feudel: Convection Patterns in Spherical Shells

gravity with r^{-5} dependence, like dielectrophoretic force of GeoFlow experiment on International Space Station

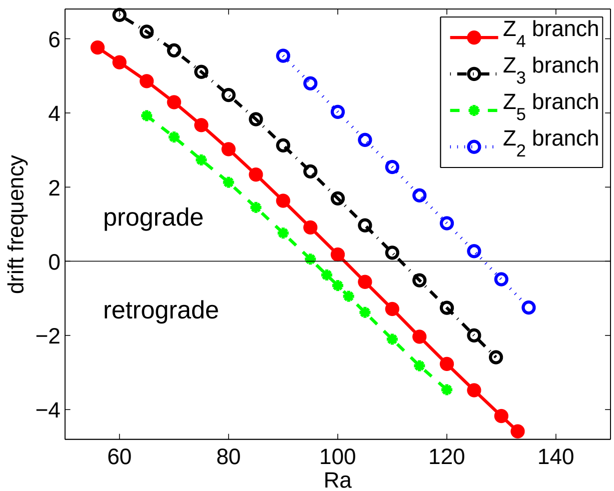


with Feudel: add rotation (break symmetry $\phi \to -\phi$)

gravity with r dependence, as in interior of constant-density earth



Rotating waveforms

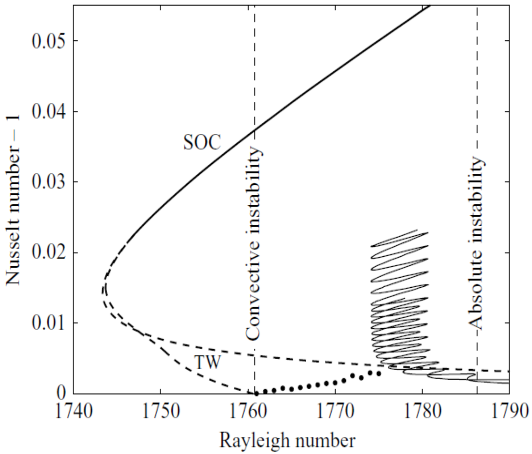


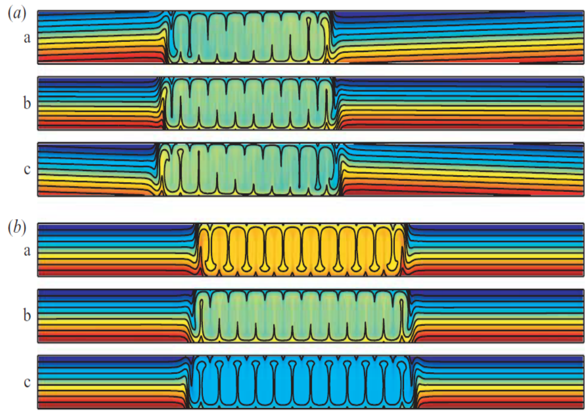
Wavespeed dependence on Ra

Snaking in 2D Binary Fluid Convection:

Mercader et al, Barcelona

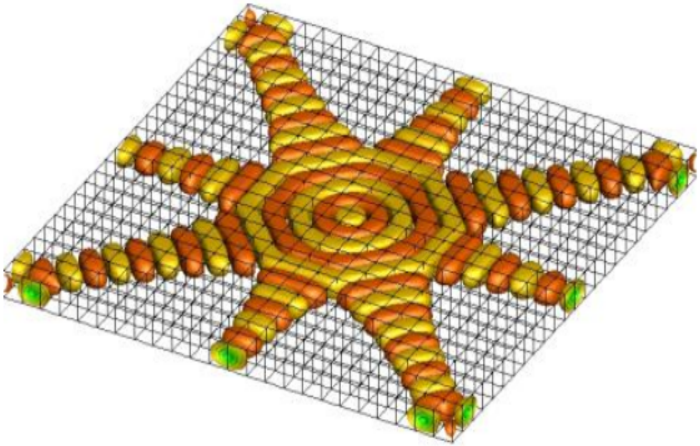
O. Batiste, E. Knobloch, A. Alonso and I. Mercader





Snaking in 2D Binary Mixture Porous Medium Convection:

Bergeon et al, Toulouse



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,

$$\left.\begin{array}{l}\textbf{Each product } u \to Au \textbf{ requires } M^2 \textbf{ operations}\\ \textbf{Generating } M \textbf{ eigenpairs requires } M \textbf{ iterations}\end{array}\right\} M^3$$

Can gain:

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

Aim method at desired eigenvalues.

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

$$u \rightarrow Au \rightarrow A^2u \rightarrow A^3u \rightarrow A^4u \rightarrow A^5u \rightarrow A^6u \rightarrow A^7u$$

Power Method

$A^2u \rightarrow v = \dfrac{A^n u}{\|A^n u\|}$

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

Arnoldi Decomposition

$A^4u \ldots A^7u \rightarrow \begin{bmatrix} v_1 & v_2 & \cdots & v_K \\ \downarrow & \downarrow & \downarrow & \downarrow \\ \downarrow & \downarrow & \downarrow & \downarrow \end{bmatrix}$

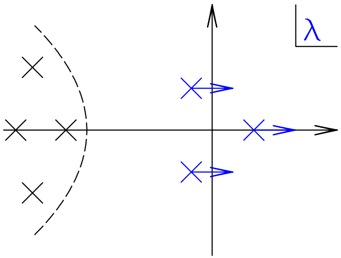
$\underset{K\times K}{H} = \underset{K\times N}{V^T} \underset{N\times N}{A} \underset{N\times K}{V} \qquad H_{ij} \equiv \langle v_i, Av_j \rangle$

$\underset{N\times N}{A} \approx \underset{N\times K}{V} \underset{K\times K}{H} \underset{K\times N}{V^T} \qquad AV \approx VH$

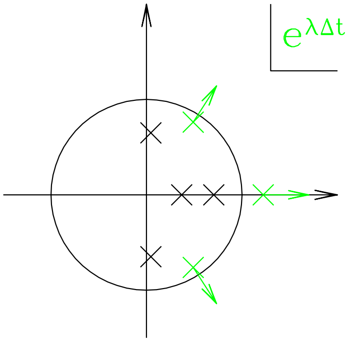
$H\phi = \lambda\phi \qquad AV\phi \approx VH\phi$

$= \lambda V\phi$

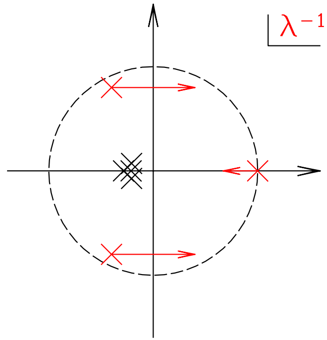
Matrix Transformations



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



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



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

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

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

Exponential Power Method

$$\boldsymbol{u}_{n+1} = (\boldsymbol{I} - \boldsymbol{\Delta} \boldsymbol{t} \boldsymbol{L})^{-1} (\boldsymbol{I} + \boldsymbol{\Delta} \boldsymbol{t} \boldsymbol{N}_U) \boldsymbol{u}_n \approx \boldsymbol{e}^{\boldsymbol{\Delta t}(\boldsymbol{L}+\boldsymbol{N}_U)} \boldsymbol{u}_n$$

Approximation valid for $\Delta t \ll 1$

Time-stepping linearized evolution equation

Enhancement factor at each iteration is

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

Inverse Power Method

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

Stokes preconditioning: $$(L + N_U) u_{n+1} = u_n$$

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

$$(I - \Delta t L)^{-1} \left[I + \Delta t N_U - (I - \Delta t L) \right] u_{n+1} = (I - \Delta t L)^{-1} \Delta t u_n$$

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

Solve with Conjugate Gradient (Bi-CGSTAB) method.

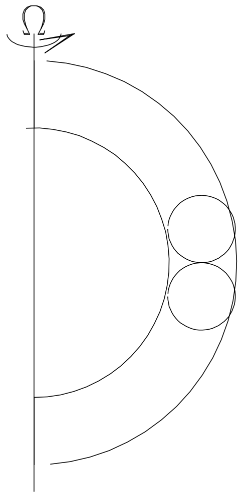
Enhancement factor at each iteration is $\left| \frac{\lambda_2}{\lambda_1} \right| \gg 1$ **for** $\lambda_1 \approx 0$

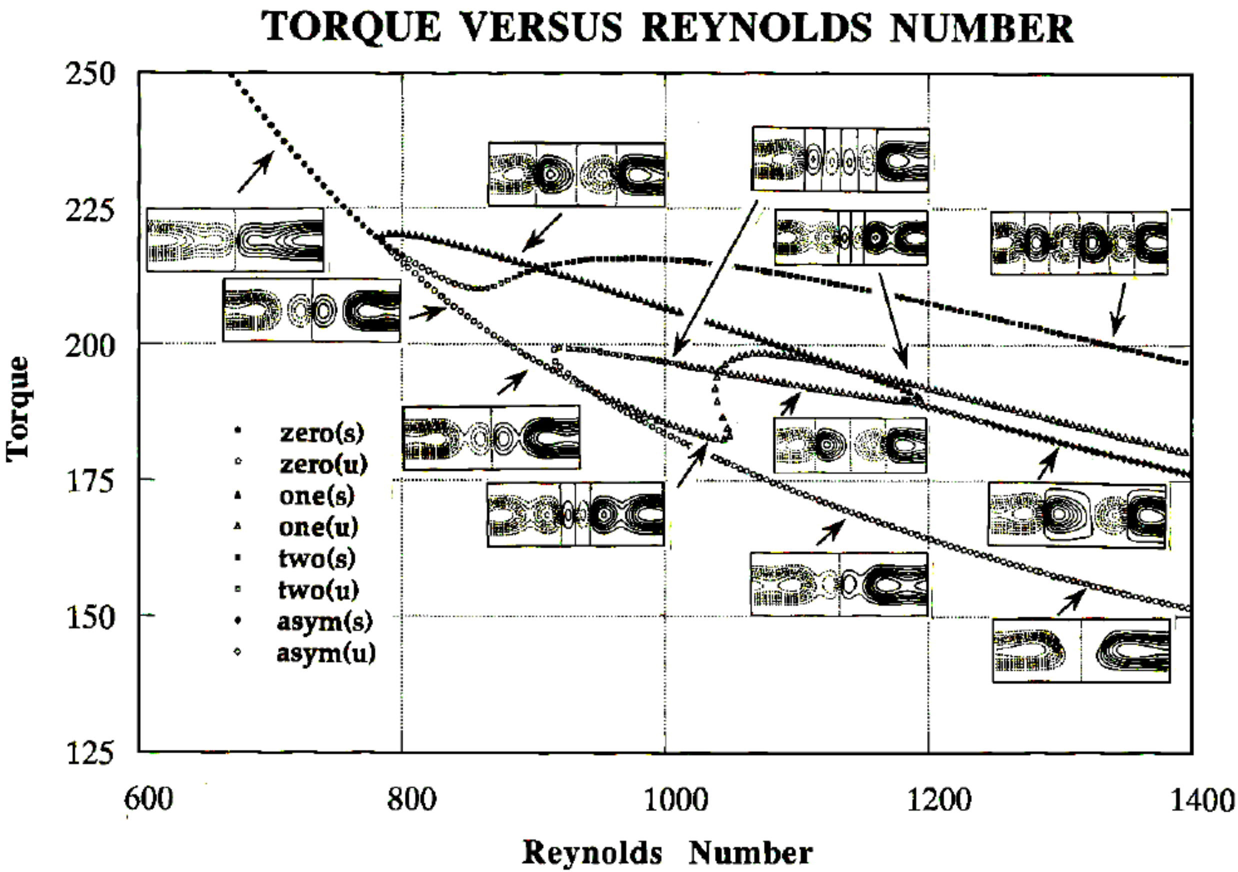
Can shift to find eigenvalues closest to s

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

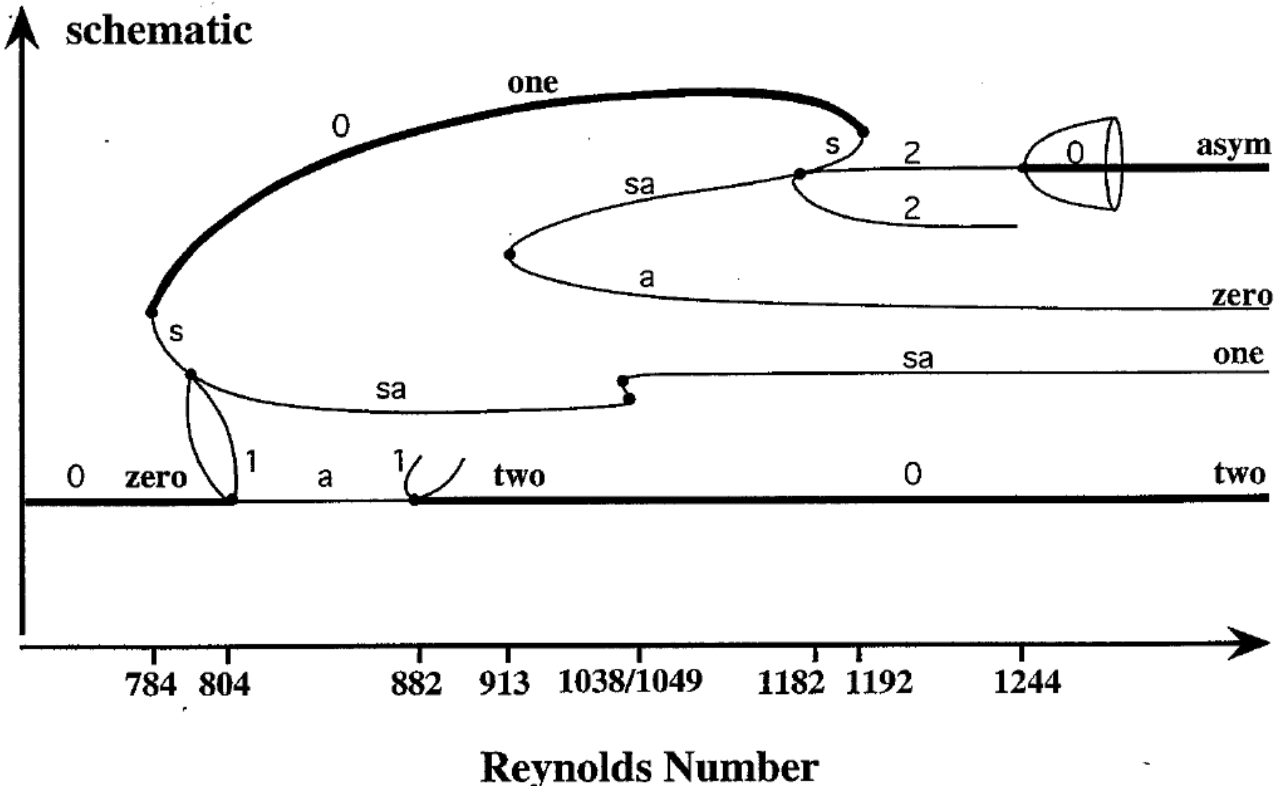
AXISYMMETRIC SPHERICAL COUETTE FLOW WITH $(r_2 - r_1)/r_1 = 0.18$

with C.K. Mamun



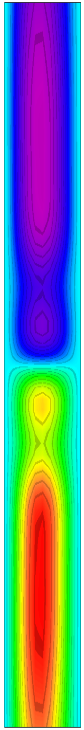

TORQUE VERSUS REYNOLDS NUMBER
Torque
Reynolds Number
250
225
200
175
150
125
600
800
1000
1200
1400
zero(s)
zero(u)
one(s)
one(u)
two(s)
two(u)
asym(s)
asym(u)

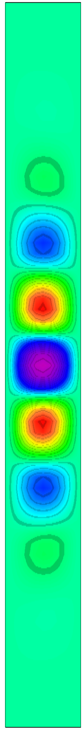
AXISYMMETRIC SPHERICAL COUETTE FLOW



Axisymmetric Spherical Couette Flow

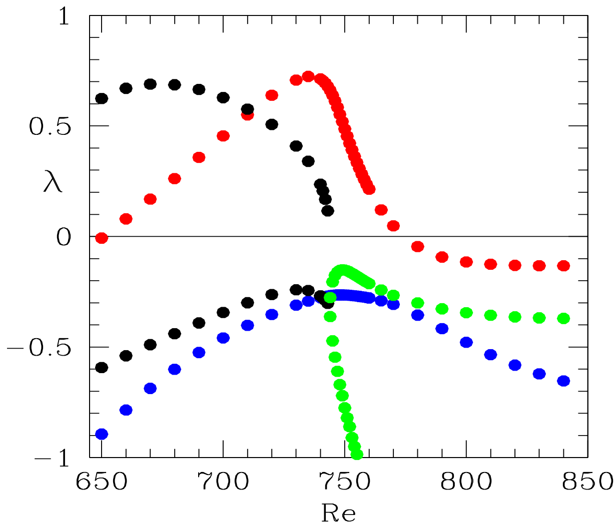
Basic flow at $Re = 650$

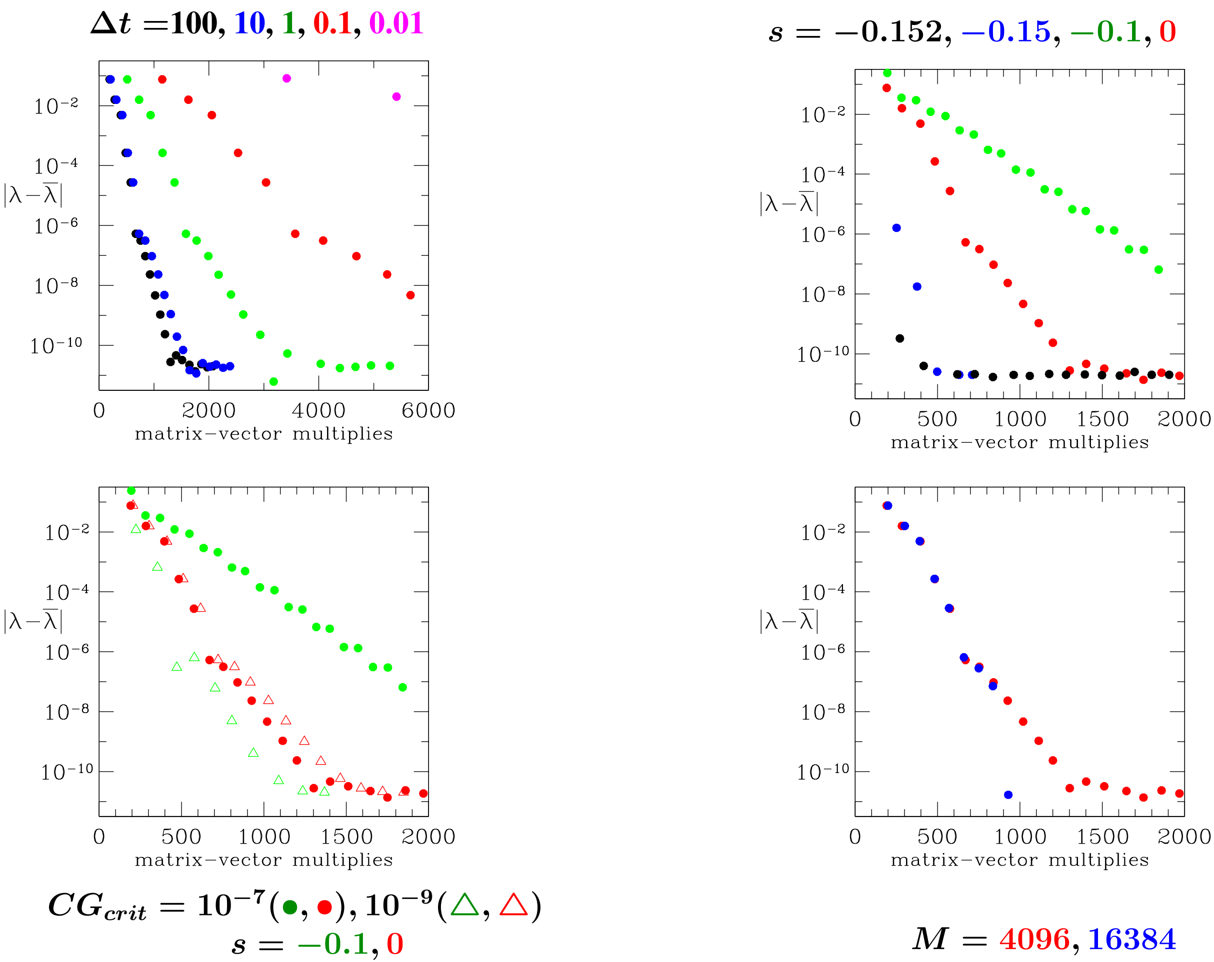




Leading eigenvector

Inverse Power Method on Spherical Couette Flow





Δt =100, 10, 1, 0.1, 0.01
s = −0.152, −0.15, −0.1, 0
|λ−λ̄|
10−2
10−4
10−6
10−8
10−10
0
2000
4000
6000
500
1000
1500
matrix−vector multiplies
CGcrit = 10−7(●, ●), 10−9(△, △)
s = −0.1, 0
M = 4096, 16384

Optimal Forcing

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

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

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

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

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 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

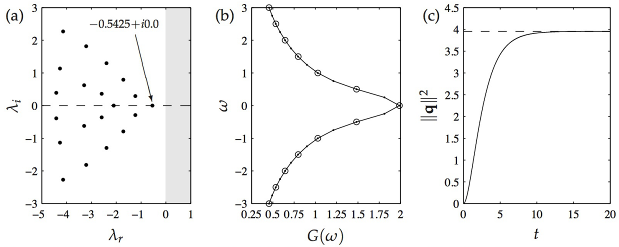


Figure 6: Frequency response of the lid-driven cavity at $Re = 100$. The eigenvalue spectra showing the 20 leading eigenvalues of the lid-driven cavity is plotted in frame (a) (the region of instability is colored in gray). The energy amplification for different frequencies is shown in frame (b), where 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

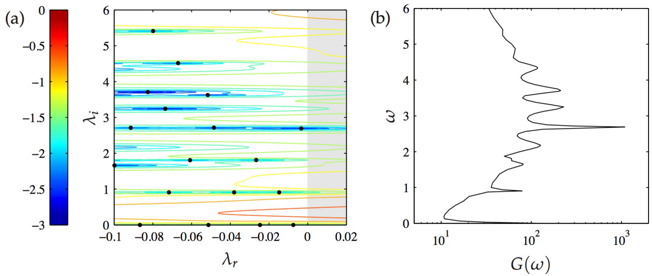


Figure 7: Frequency response of the lid-driven cavity at $Re = 8015$. The leading eigenvalues of the lid-driven cavity and the ε-pseudospectrum (logarithmically spaced coloured contours) are plotted in frame (a) (the region of instability is colored in gray). The energy amplification for different frequencies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

Forcing

Response

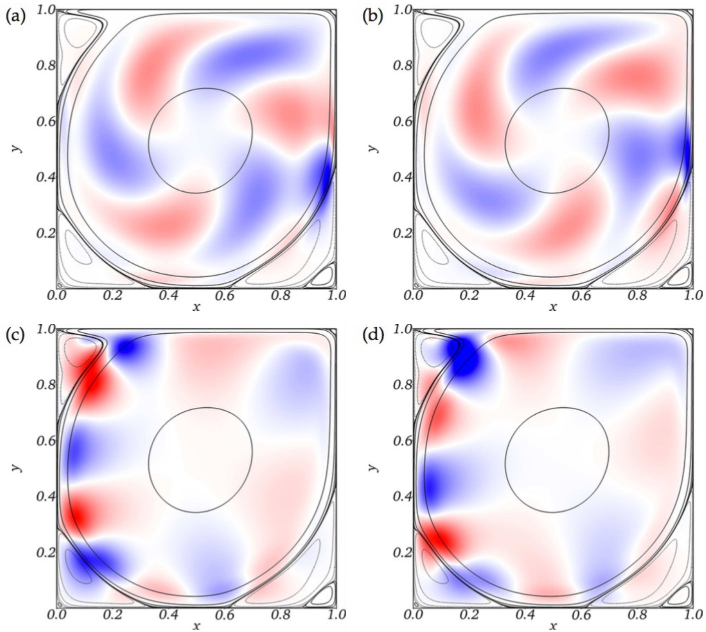


Figure 8: Optimal forcing profile at $\omega = 2.6875$ for the two-dimensional lid-driven cavity, Re=8015. Real and imaginary parts of the optimal forcing profile are shown in (a) and (b), respectively, together with real and imaginary part of the resulting flow response shown in (c) and (d). The color 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

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**

BOSE-EINSTEIN CONDENSATION

Ultra-cold coherent state of matter

Predicted by Bose (1924) and Einstein (1925)

Realized experimentally by Cornell, Ketterle, Wieman (1995)

Nobel prize (2001)

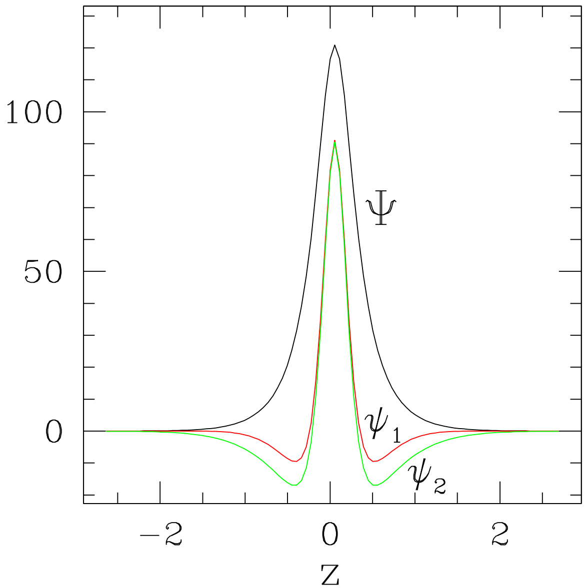
Gross-Pitaevskii / Nonlinear Schrödinger Equation

$$\partial_t \Psi = i\, [\underbrace{\frac{1}{2}\nabla^2}_{L} + \underbrace{\mu - V(r) - a|\Psi|^2}_{N}]\Psi$$

$$V(\mathrm{x}) = \frac{1}{2}|\omega \cdot \mathrm{x}|^2 = \frac{1}{2}(\omega_r r^2 + \omega_z z^2) \qquad \text{(cylindrical trap)}$$

Spatial discretisation up to $M = 10^2 \times 10^2 \times 10^2 = 10^6$

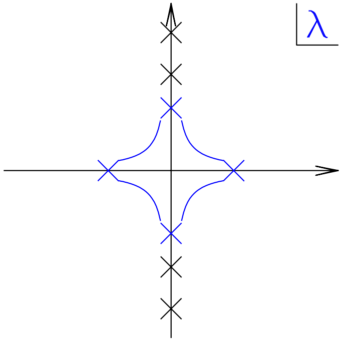
Eigenvalues, energies determine decay rates of condensate.



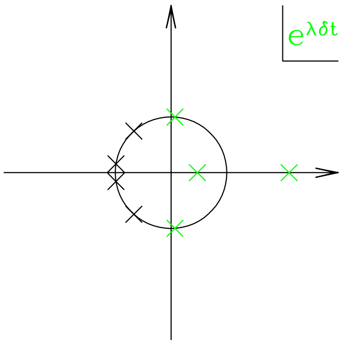
100
50
0
−2
0
2
Z
Ψ
ψ1
ψ2

Hamiltonian Systems

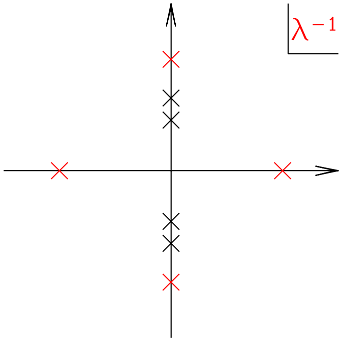
$$f(A) = A$$



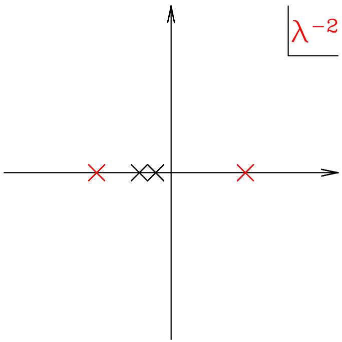
$$f(A) = e^{A\Delta t}$$



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



$$f(A) = A^{-2}$$



STEADY STATE SOLVING:

$$0 = L\Psi + N(\Psi)$$

Newton's method + Stokes preconditioning + BICGSTAB

LINEAR STABILITY OF STEADY STATE Ψ:

$$\partial_t \psi = i\,[(\frac{1}{2}\nabla^2 + \mu - V(r))\psi - a\Psi^2(2\psi + \psi^*)]$$

$$A \begin{pmatrix} \psi^R \\ \psi^I \end{pmatrix} \equiv \begin{bmatrix} 0 & -(L + DN^I) \\ L + DN^R & 0 \end{bmatrix} \begin{pmatrix} \psi^R \\ \psi^I \end{pmatrix}$$

$$\begin{aligned} DN^R &\equiv \mu - V(\mathrm{x}) - 3a\Psi^2 \\ DN^I &\equiv \mu - V(\mathrm{x}) - a\Psi^2 \end{aligned}$$

$$
A^2 \begin{pmatrix} \psi^R \\ \psi^I \end{pmatrix} =
$$

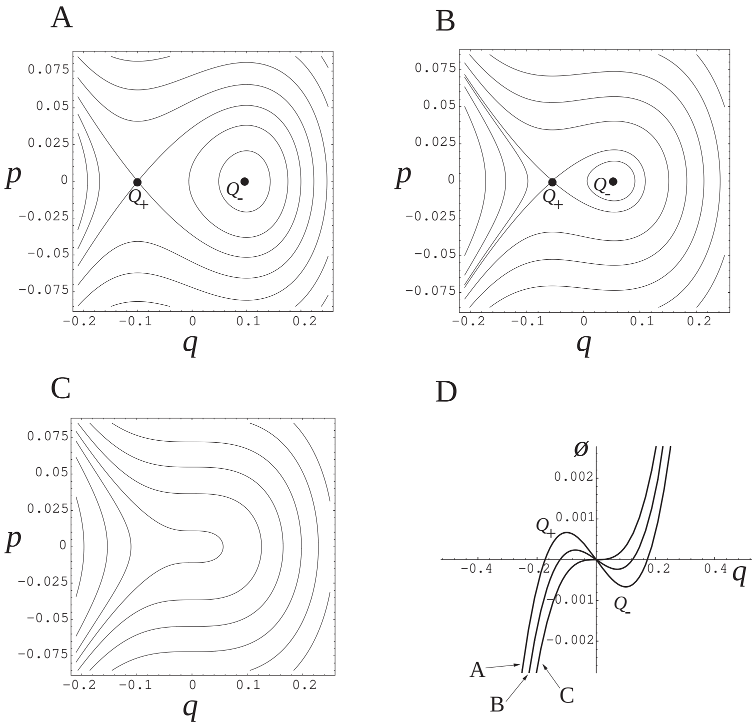
$$
\begin{bmatrix} -(L + DN^I)(L + DN^R) & 0 \\ 0 & -(L + DN^R)(L + DN^I) \end{bmatrix} \begin{pmatrix} \psi^R \\ \psi^I \end{pmatrix}
$$

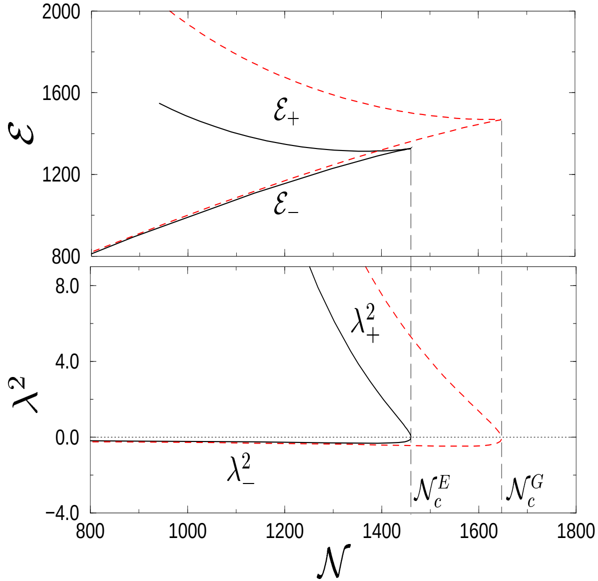
Inverse square power with Stokes preconditioning and shift:

$$
\begin{aligned}
(A^2 - s^2 I)\psi_{n+1} &= \psi_n \\
L^{-2}(A^2 - s^2 I)\psi_{n+1} &= L^{-2}\psi_n
\end{aligned}
$$

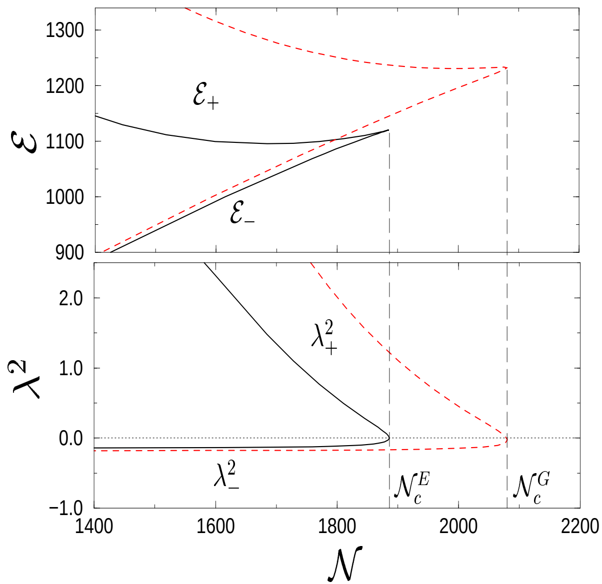
Solve with BiCGSTAB

Hamiltonian saddle-node bifurcation of hyperbolic and elliptic fixed points





$\omega_z = \omega_r/5$ **(cigar)**



$\omega_r = \omega_z/5$ **(pancake)**

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