Bifurcation Analysis for Time Steppers

Laurette Tuckerman

laurette@limsi.fr

THE THREE TOOLS OF COMPUTATIONAL FLUID DYNAMICS

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

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

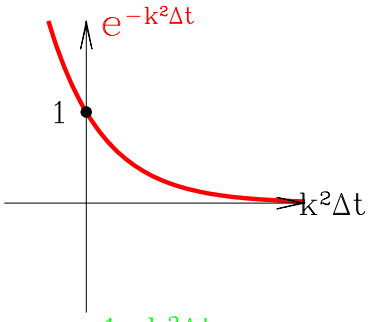
Linear stability analysis: $\lambda u = Lu + N_U u$

Heat Equation

$$\begin{aligned} \partial_t u &= \partial^2_{xx} u \\ u(x,t) &= \sum_{k=1}^{k_{max}} u_k(t) \sin kx \\ \partial_t u_k &= -k^2 u_k \end{aligned}$$

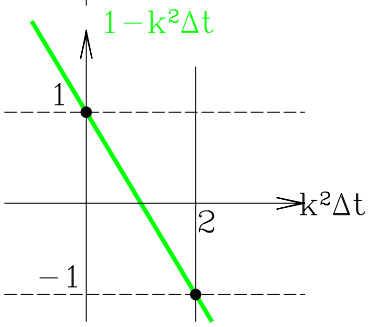
Exact solution

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



Explicit Euler

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



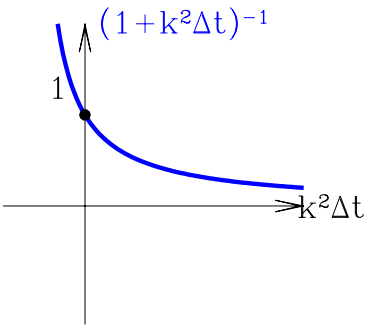
As $k_{max} \to \infty$, $\Delta t_{max} = \frac{2}{k^2_{max}} \to 0$

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

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

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

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

Explicit/Implicit Euler:

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

- **Steady state solving**

$$0 = N(U) + L\,U$$

Newton's method:

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

- **Linear stability analysis:**

$$\lambda u = N_U\,u + L\,u \equiv A_U u$$

Arnoldi/block power method:

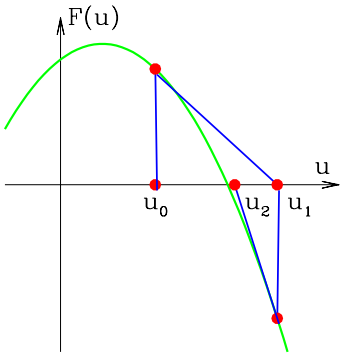
$$u_{n+1} = A_U^{-1} u_n \quad \textbf{or} \quad u_{n+1} = e^{A_U \Delta t} u_n$$

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

STEADY STATE SOLVING

$$0 = F(U)$$

Newton's method



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

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

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 $u \rightarrow Au$,

For an arbitrary matrix,

Each product $u \rightarrow Au$ requires M^2 operations
Convergence requires M iterations } M^3

Can gain:

If A is structured or sparse, then $u \rightarrow Au$ 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

$$\begin{aligned} Au &= v \\ PAu &= Pv \end{aligned}$$

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:

$$\begin{aligned} A_U u &= Lu + N_U u \\ &= \nabla^2 U - (U \cdot \nabla)u - (u \cdot \nabla)U \end{aligned}$$

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 $P \tilde{=} L^{-1}$

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 only algebraic role, and can (should) be taken large (**$\gtrsim 10^2$**).**
- Δt **interpolates between** $P = I$ **and** $P = L^{-1}$**.**
- **Called Stokes preconditioning**

One Newton Step

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

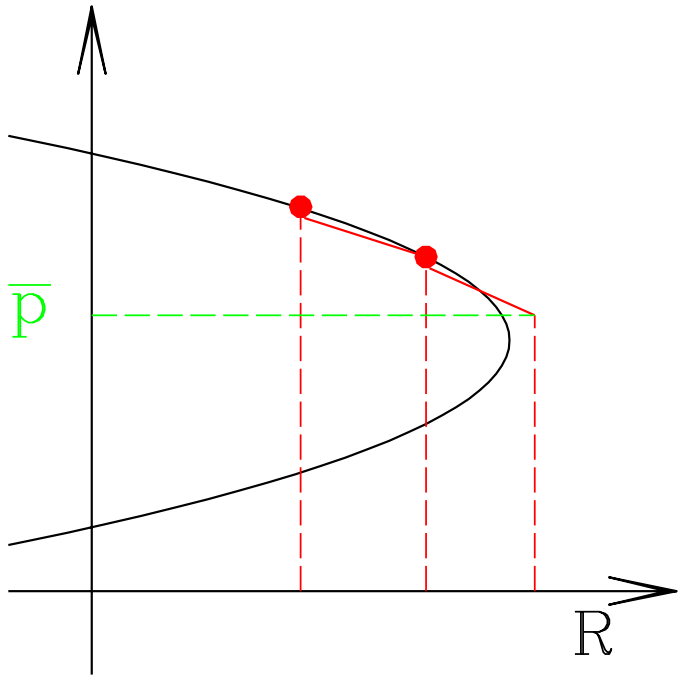
$$\underbrace{\left[(I - \Delta t L)^{-1}(I + \Delta t N_U) - I\right] u}_{\text{difference between two widely spaced consecutive linearized timesteps}} = \underbrace{\left[(I - \Delta t L)^{-1}(I + \Delta t N) - I\right] U}_{\text{difference between two widely spaced consecutive timesteps}}$$

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

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

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

Newton step

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

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

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

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).

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

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

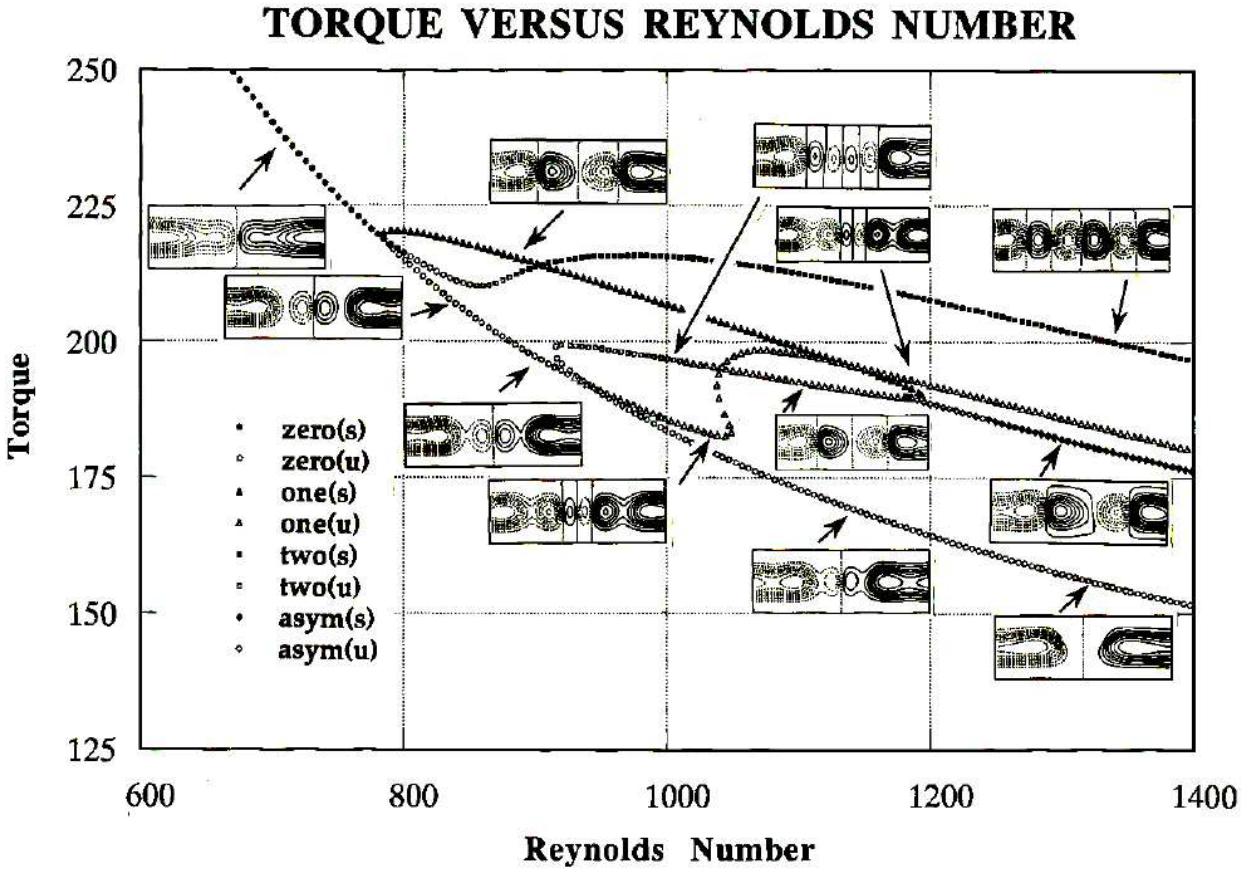
} **Use only vectors and operators of length M**

TRAVELING WAVES: $U(x - Ct, y, z)$

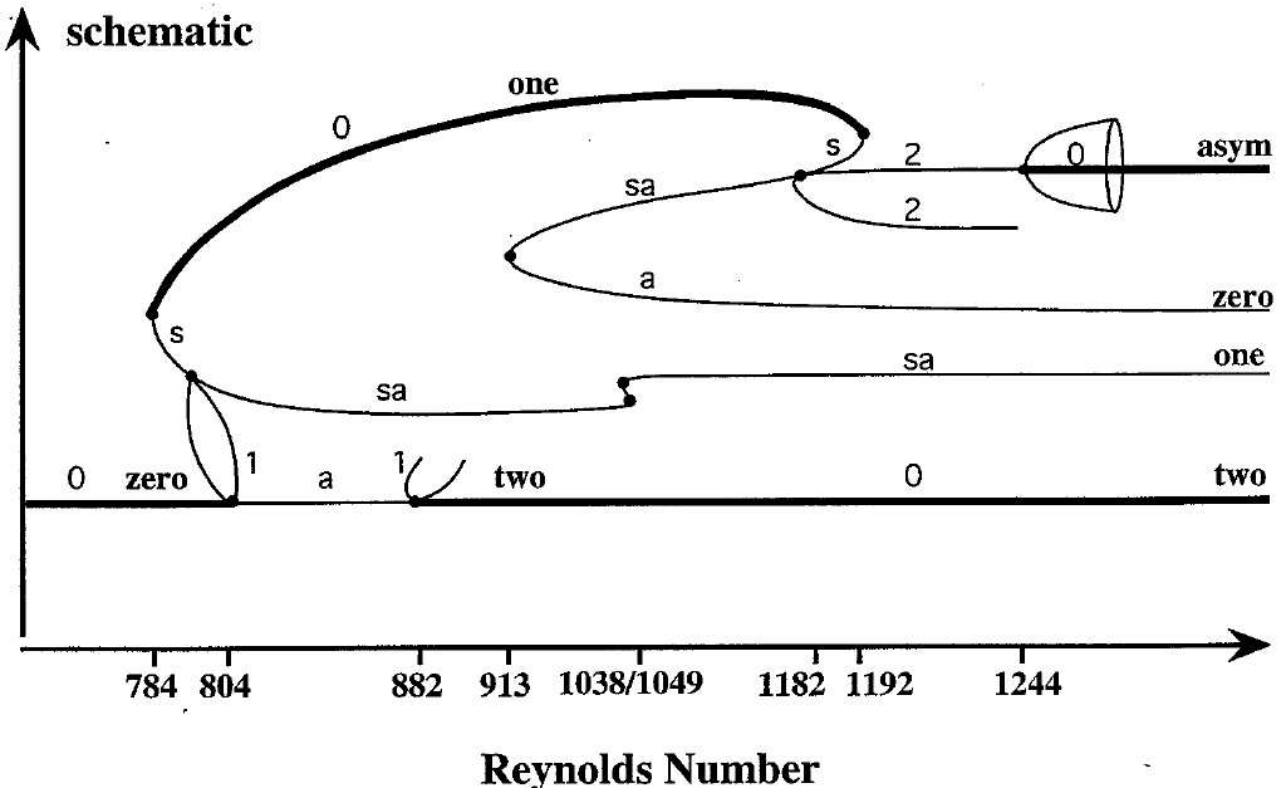
$$\text{Goal} \begin{cases} 0 = C\partial_x U + N(U) + LU \\ 0 = p(U) - \bar{p} \end{cases}$$

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

AXISYMMETRIC SPHERICAL COUETTE FLOW WITH $\sigma = 0.18$



AXISYMMETRIC SPHERICAL COUETTE FLOW WITH $\sigma = 0.18$



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 $u \rightarrow Au$

To diagonalise an arbitrary matrix,

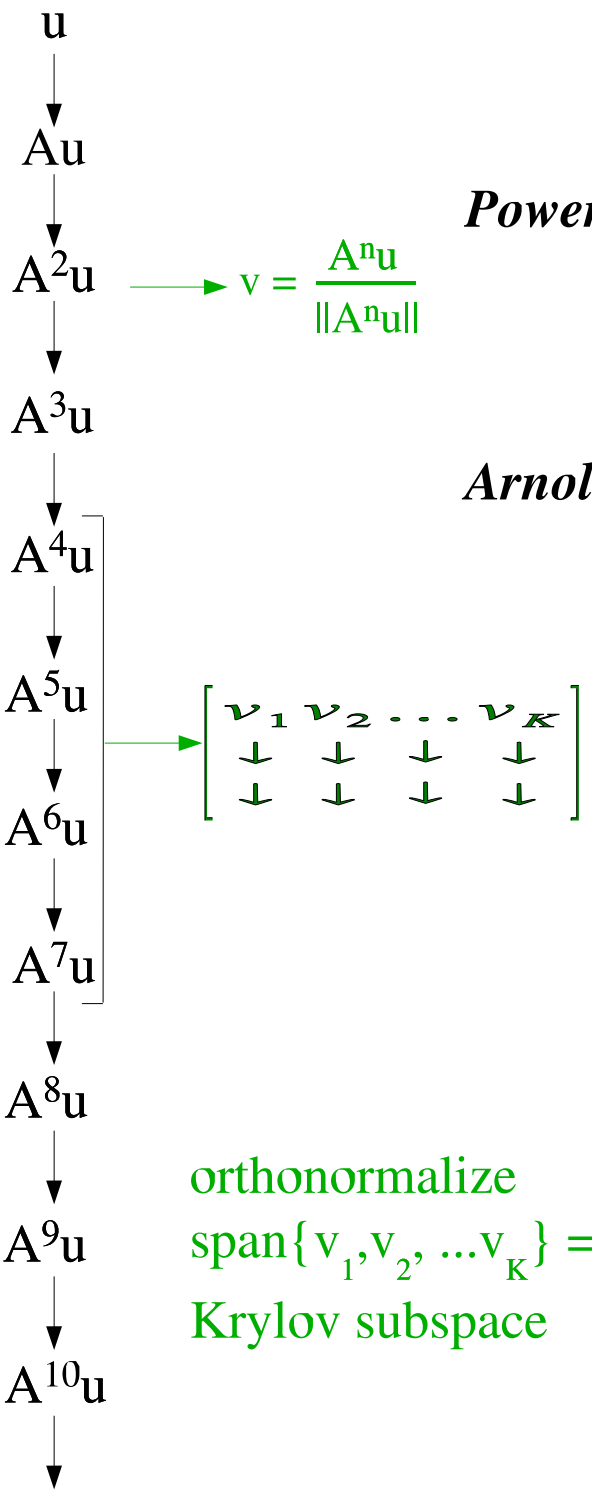
Each product $u \rightarrow Au$ requires M^2 operations
Generating M eigenpairs requires M iterations } M^3

Can gain:

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

Aim method at desired eigenvalues.

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



Power Method

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

Arnoldi Decomposition

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

(K×K = K×N · N×N · N×K)

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

(N×N ≈ N×K · K×K · K×N)

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

orthonormalize

span$\{v_1, v_2, \ldots v_K\}$ = Krylov subspace

H (Hessenberg matrix)

$K \approx 4$

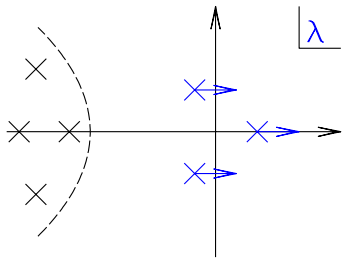
$N \approx 10^6$

MATRIX TRANSFORMATIONS

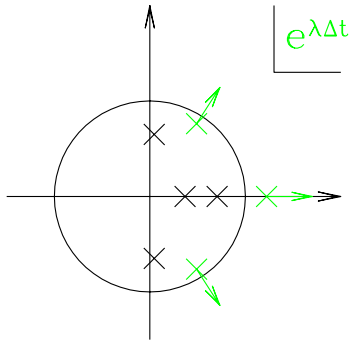
If $A\,u = \lambda\,u$

then $f(A)\,u = f(\lambda)\,u$

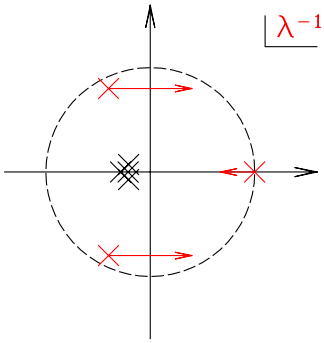
$f(A) = A$



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



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



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

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

EXPONENTIAL POWER METHOD

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

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

$$\left|\frac{\lambda_2}{\lambda_1}\right| \gg 1 \qquad \text{for } \lambda_1 \approx 0$$

Can shift to find eigenvalues closest to s

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

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)

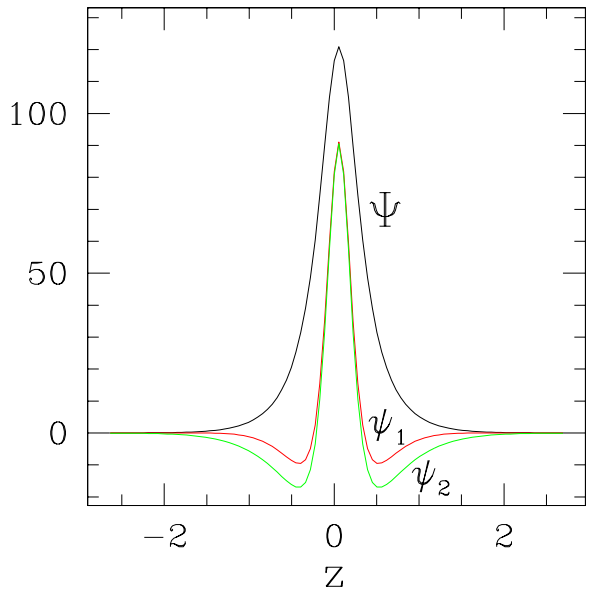
Gross-Pitaevskii / Nonlinear Schrödinger Equation

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

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

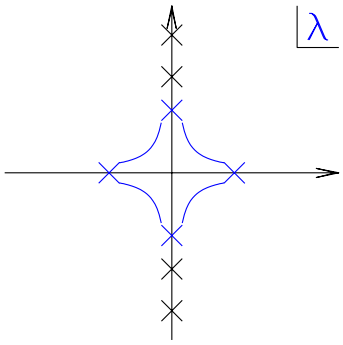
Spatial discretisation up to $M = 10^2 \times 10^2 \times 10^2 = 10^6$

Eigenvalues, energies determine decay rates of condensate.

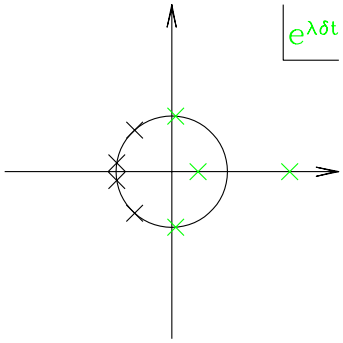


Hamiltonian Systems

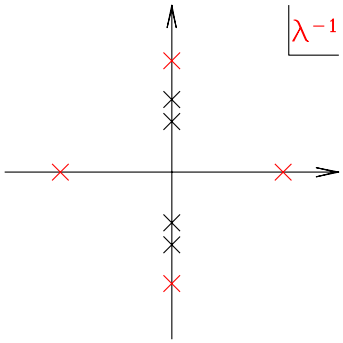
$f(A) = A$



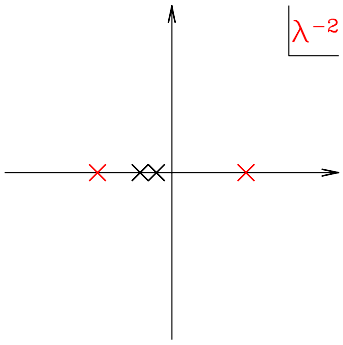
$f(A) = e^{A\Delta t}$



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



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



STEADY STATE SOLVING:

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

Newton's method + Stokes preconditioning + BICGSTAB

LINEAR STABILITY OF STEADY STATE Ψ:

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

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

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

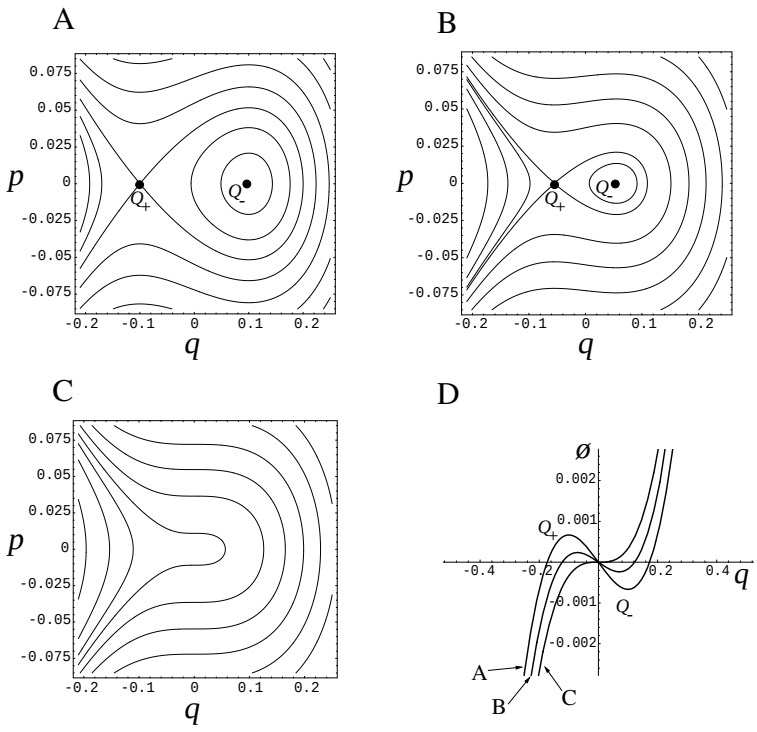
$$A^2\begin{pmatrix} \psi^R \\ \psi^I \end{pmatrix} =$$
$$\begin{bmatrix} -(L + DN^I)(L + DN^R) & 0 \\ 0 & -(L + DN^R)(L + DN^I) \end{bmatrix} \begin{pmatrix} \psi^R \\ \psi^I \end{pmatrix}$$

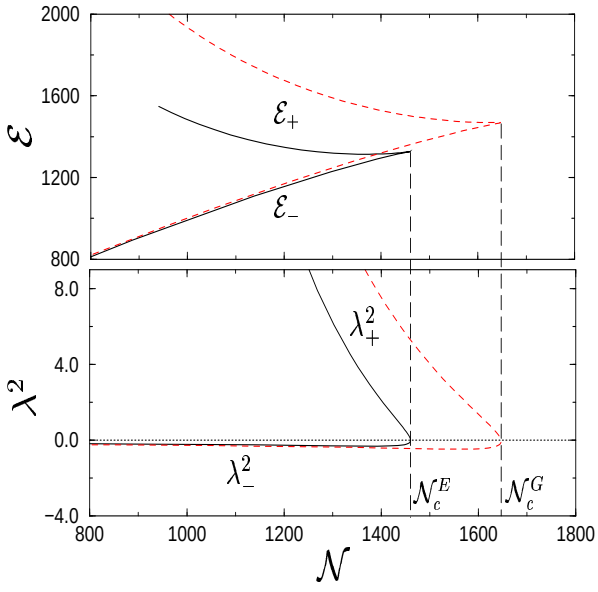
Inverse square power method
with Stokes preconditioning and shift:

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

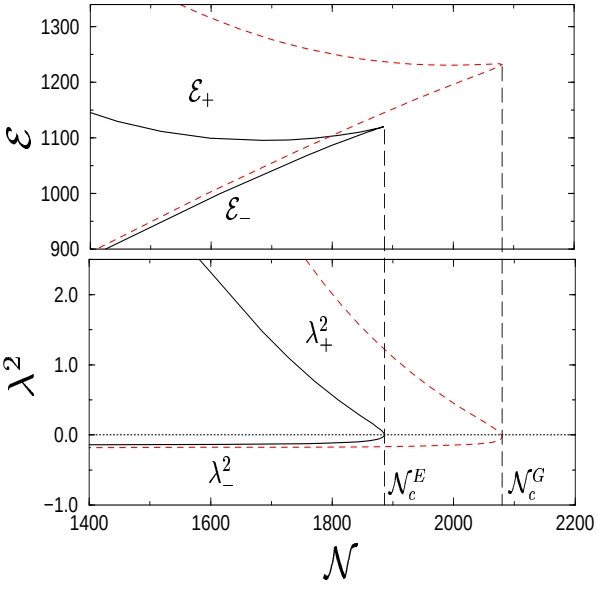
Solve with BICGSTAB

Hamiltonian saddle-node bifurcation of hyperbolic and elliptic fixed points





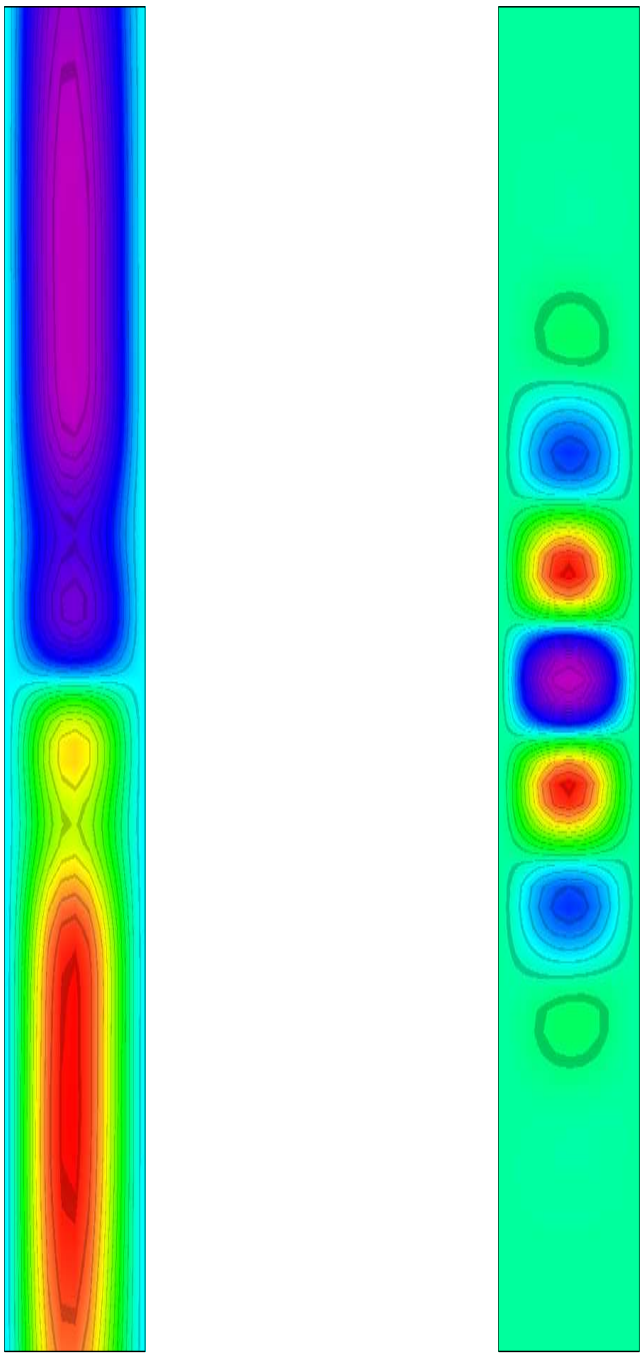
$\omega_z = \omega_r/5$ **(cigar)**



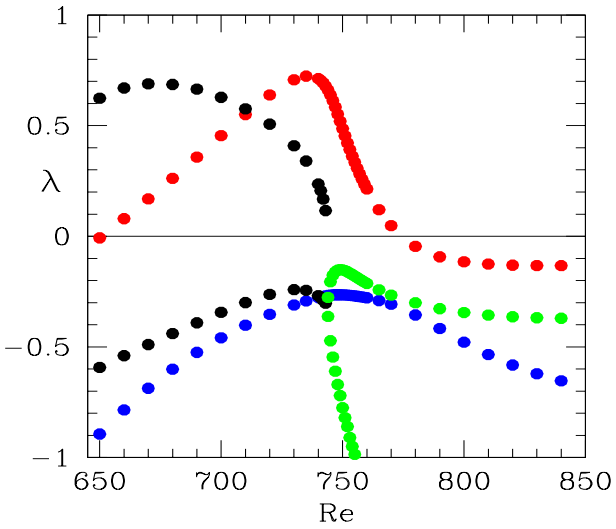
$\omega_r = \omega_z/5$ **(pancake)**

AXISYMMETRIC SPHERICAL COUETTE FLOW WITH $\sigma = 0.18$

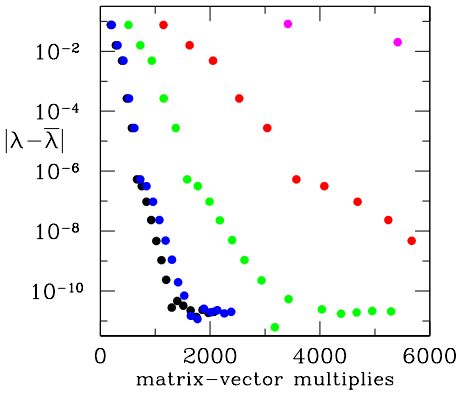
Basic flow at $Re = 650$ **Leading eigenvector**



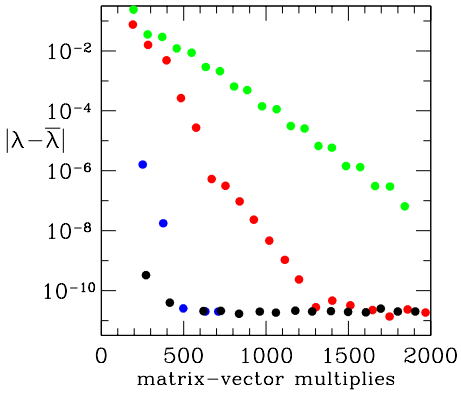
Inverse Power Method on Spherical Couette Flow

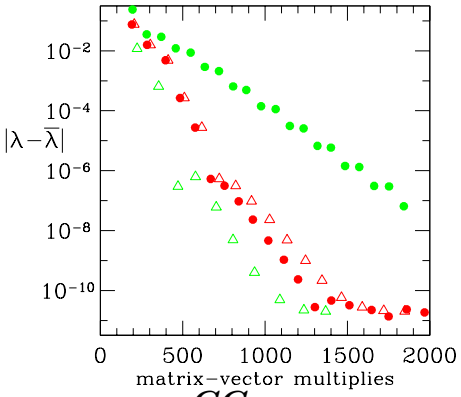


Δt =100, 10, 1, 0.1, 0.01

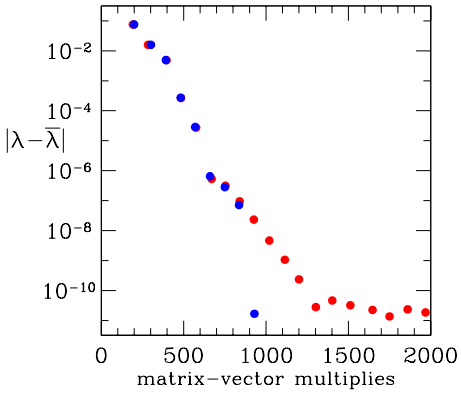


$s = -0.152, -0.15, -0.1, 0$





CG_{crit} = 10^{-7}(●, ●), 10^{-9}(△, △)
$s = -0.1, 0$



$M = 4096, 16384$

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**