Spectral Methods Without Discretization Woes

Alex Townsend
Cornell University
townsend@cornell.edu

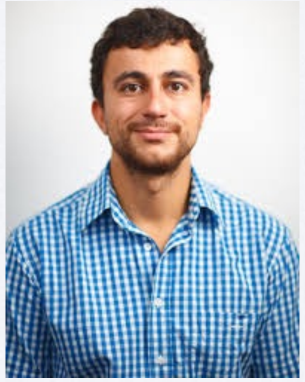
Colleagues that cure discretization woes:


Dan Fortunato


Marc Gilles


Nick Hale


Andrew Horning


Sheehan Olver


Nick Trefethen


Geoff Vasil


Grady Wright


Heather Wilber

Algorithms, adaptivity, & encapsulation

julia

ApproxFun

[Olver, Slevinsky, & others]

DEDALUS

[Burns, Vasil, Oishi, Lecoanet, & Brown]

Algorithms

Adaptivity

Encapsulation

chebfun

[Trefethen, Driscoll, Hale, & many others]

Functions

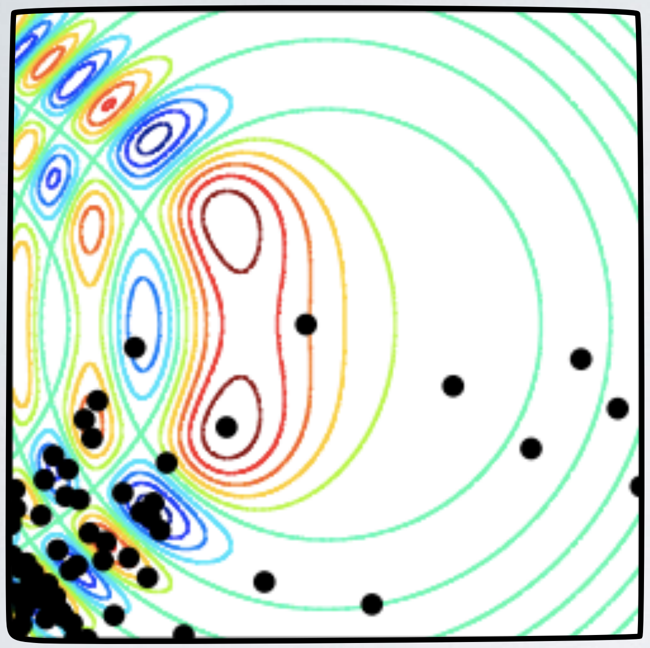
DEs

(Geometry)

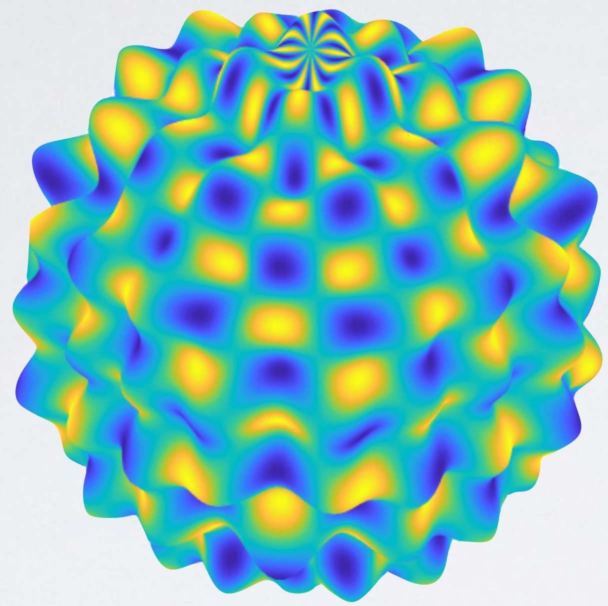
Goal: Discretization oblivious users.

Highlight reel: functions

Adaptive 2D approximation

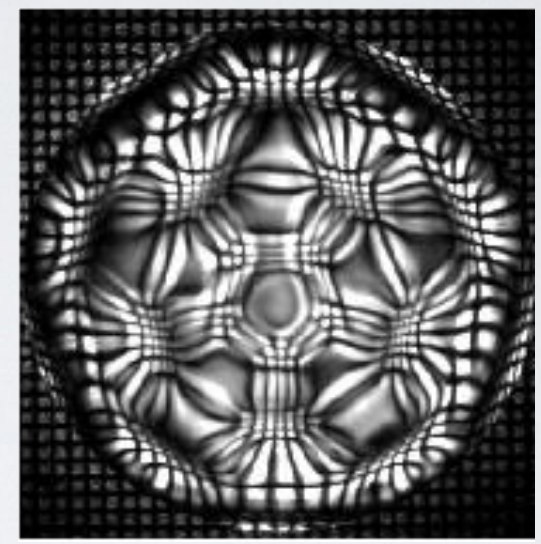


with Trefethen



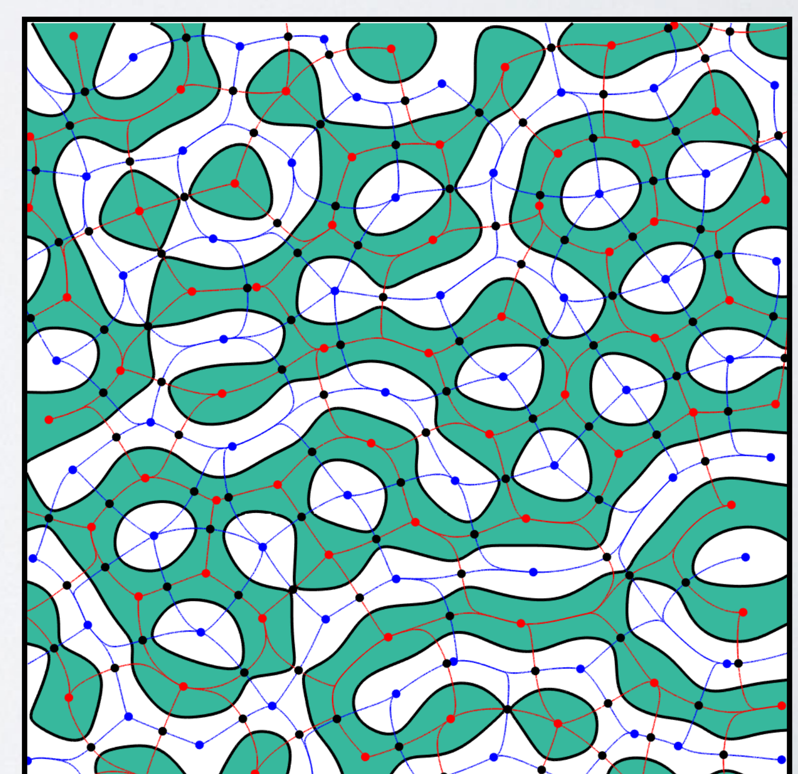
with Platte

Fast rotate on hemisphere



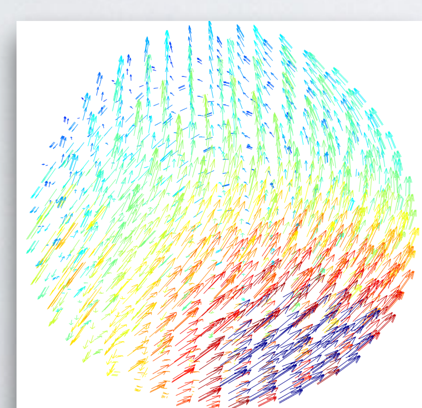
with Bostwick, Steen, & Zhao

Robust rootfinders

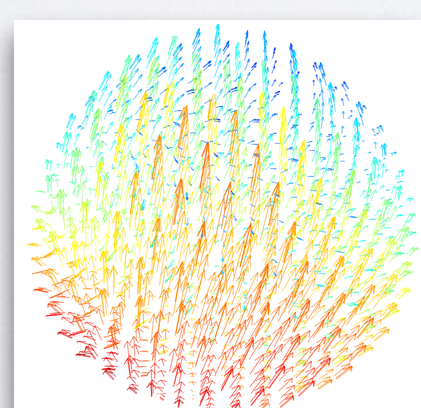


with Belyaev

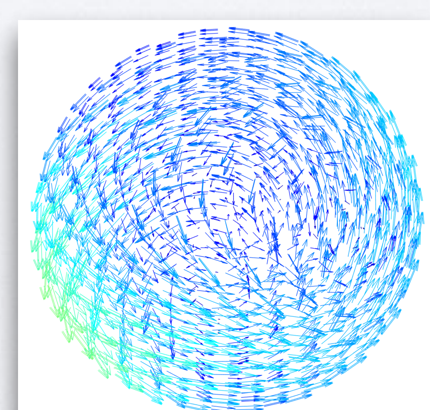
Helmholtz decomposition in ball



$=$



$+$



$$\underline{v} = \nabla f + \nabla \times \psi$$

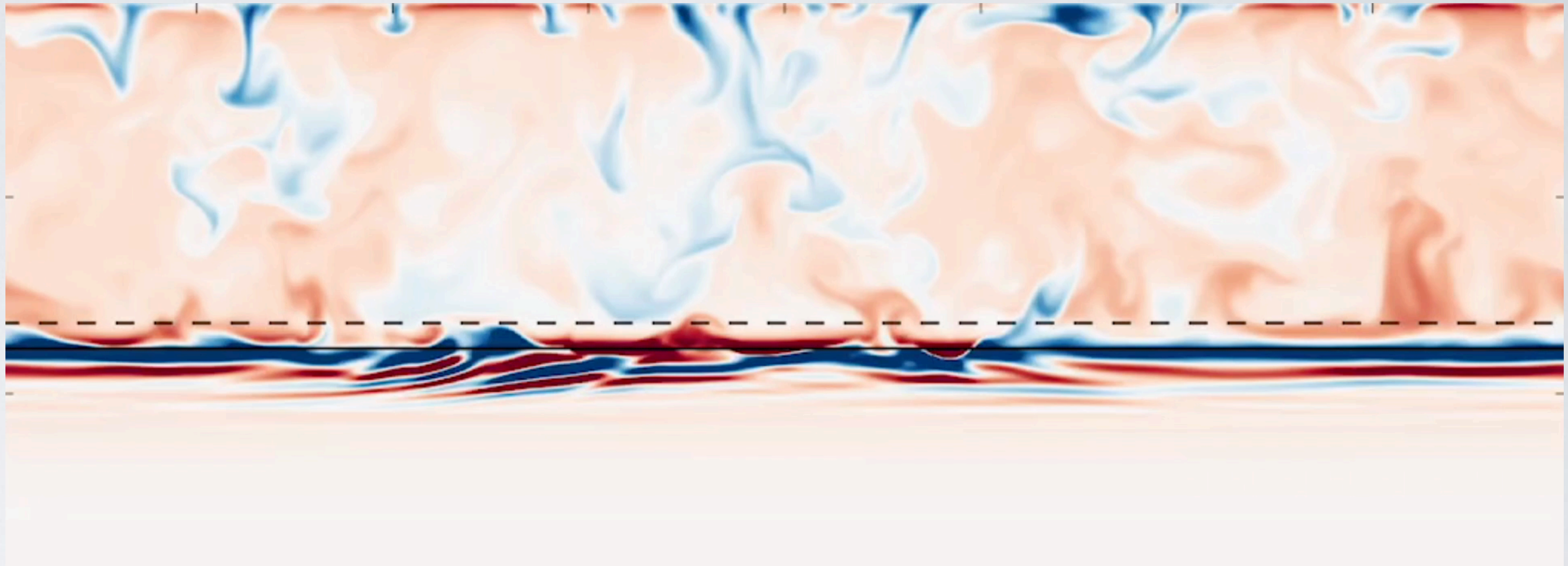
with Boulle

Highlight reel: DEs

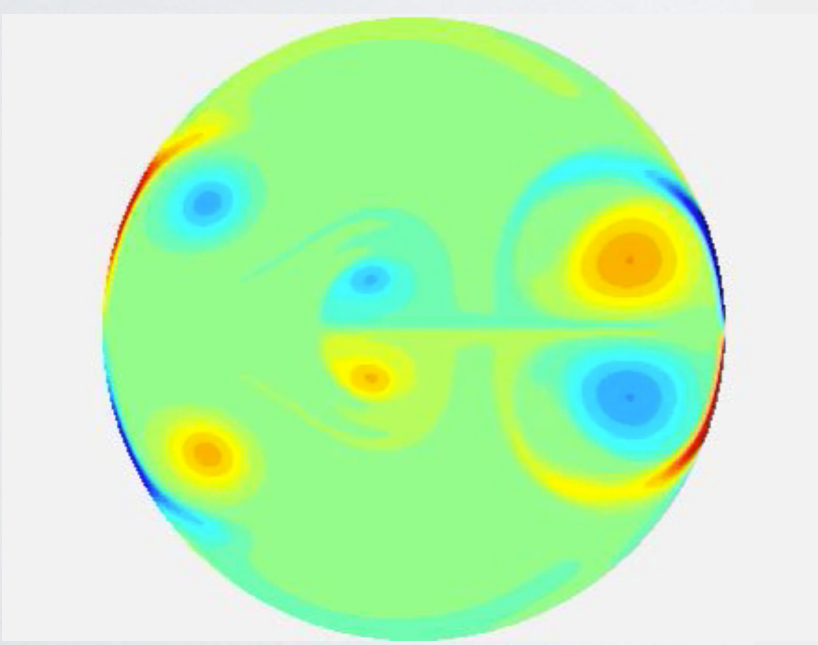
Convection waves

DEDALUS

Keaton Burns, Geoff Vasil

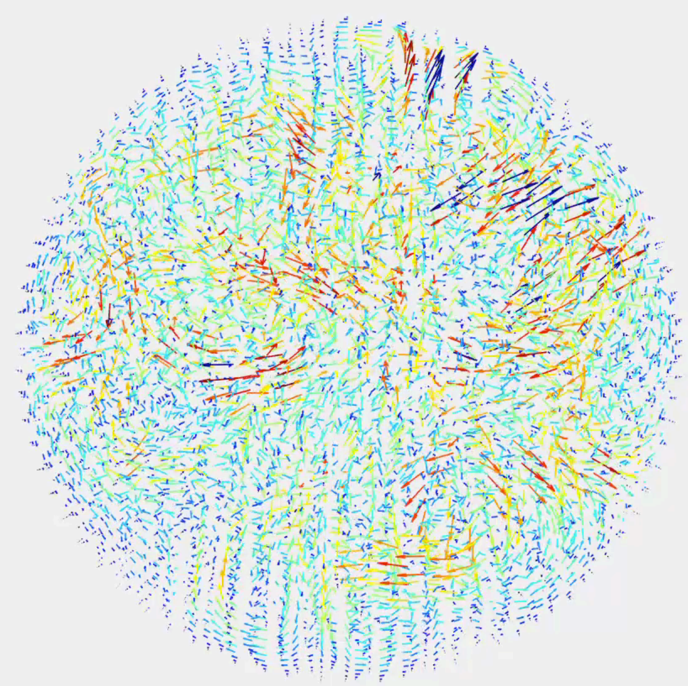


Navier-Stokes on disk



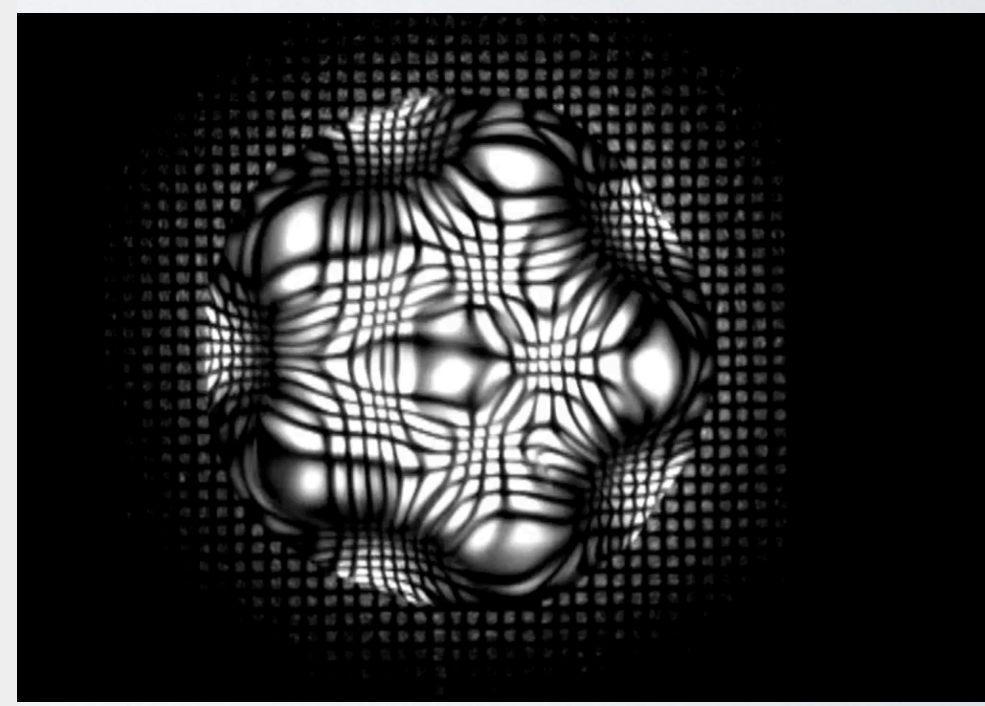
18.336, Fall 2015

Active fluids in 3D ball



with Boulle, Słomka, & Dunkel

Sessile drops (hemisphere)



with Bostwick, Steen, & Wesson

Timeline

chebfun

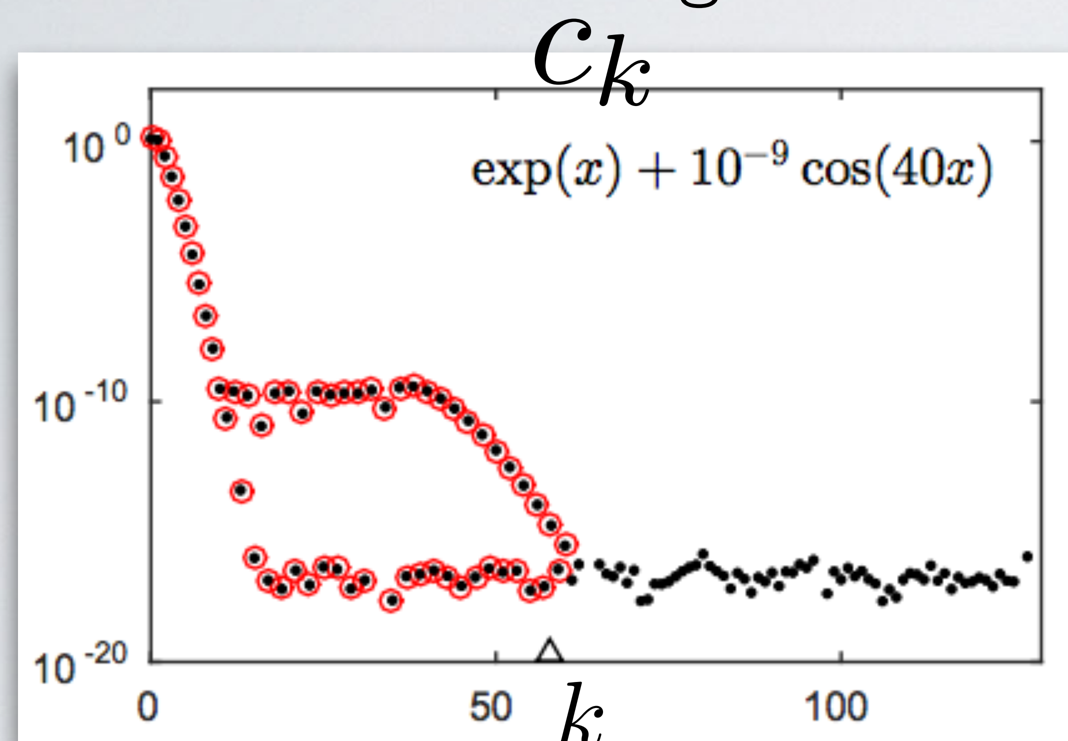
2004 2011 2018

The 1D era

The 1D era: functions

Continuous analogue of MATLAB: e.g. $\texttt{sum(f)} = \int_{-1}^{1} f(x)dx$

$$\texttt{diff(f)} = f'(x)$$



Chebyshev expansions

$$f(x) \approx \sum_{k=0}^{n} c_k T_k(x)$$

or piecewise
or weighted
or mapped

```
>> f = chebfun(@(x) abs(4*cos(3*pi*x))./(x-.2));
   chebfun column (8 smooth pieces)
       interval            length      endpoint values    endpoint exponents
[       -1,     -0.83]        13       -3.3 -7.6e-16           [0       0]
[    -0.83,      -0.5]        16   -4.8e-15 -1.6e-14           [0       0]
[     -0.5,     -0.17]        20    1.3e-14     6e-16           [0       0]
[    -0.17,      0.17]        59    2.9e-14 -2.6e-14           [0       0]
[     0.17,       0.2]         9   -2.6e-13     -Inf           [0      -1]
[      0.2,       0.5]        15        Inf -4.8e-14           [-1       0]
[      0.5,      0.83]        21    6.3e-14  1.4e-14           [0       0]
[     0.83,         1]        13   -1.3e-14        5           [0       0]
vertical scale = Inf     Total length = 166
```


Nick Hale


Rodrigo Platte

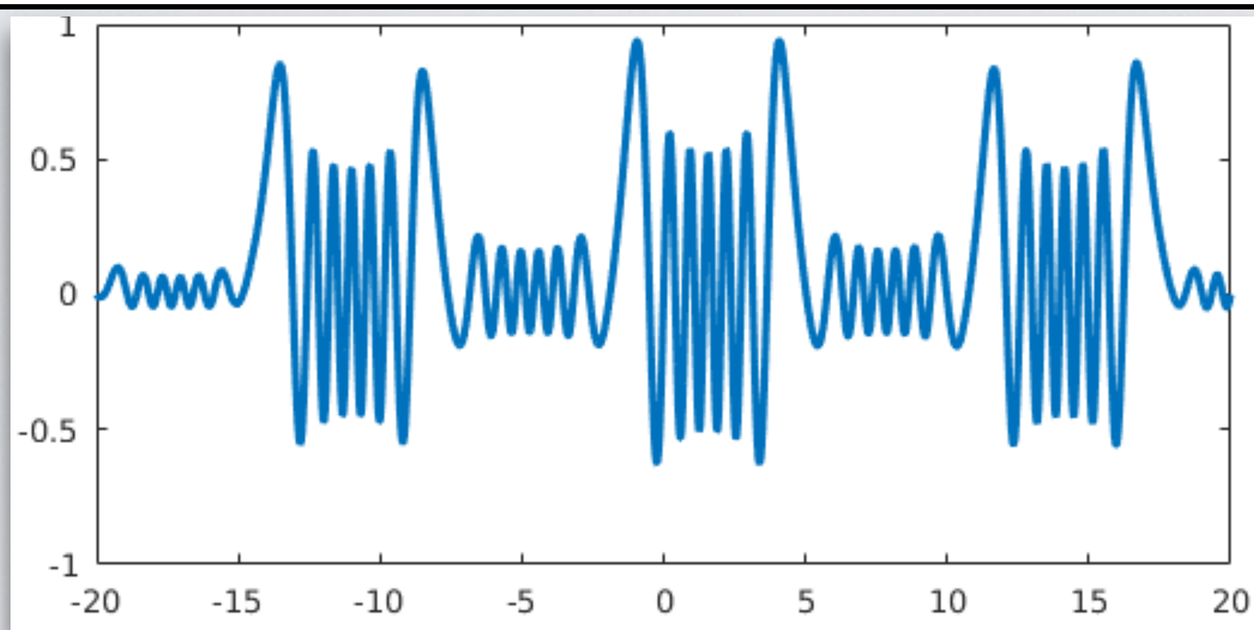

LNT


Ricardo Pachon

The 1D era: DEs

Linear and nonlinear systems of BVPs:

```
plot( chebop(@(x,u) diff(u,2)+50*(1+sin(x)).*u,[-20,20],0,0)\1 )
```

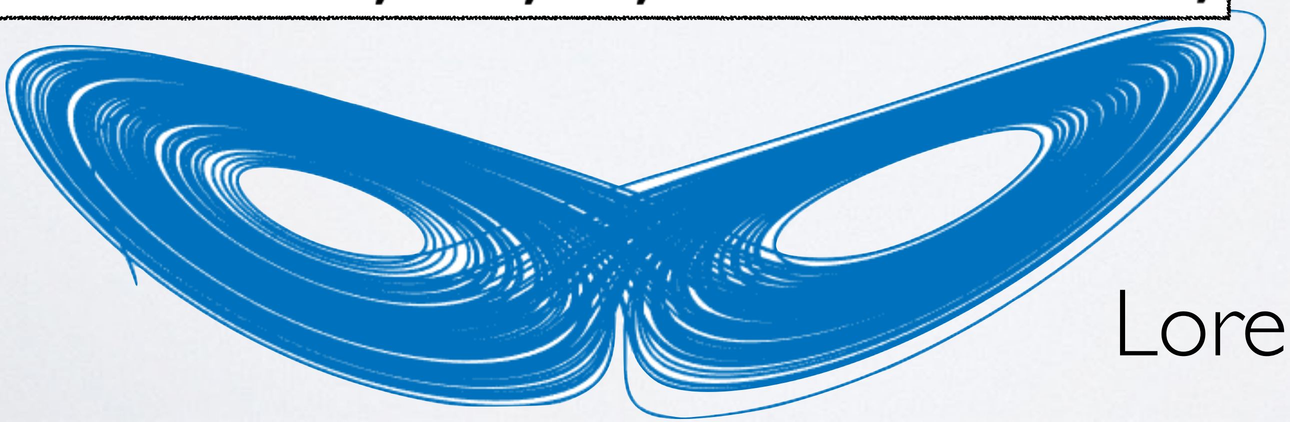


Ásgeir Birkisson

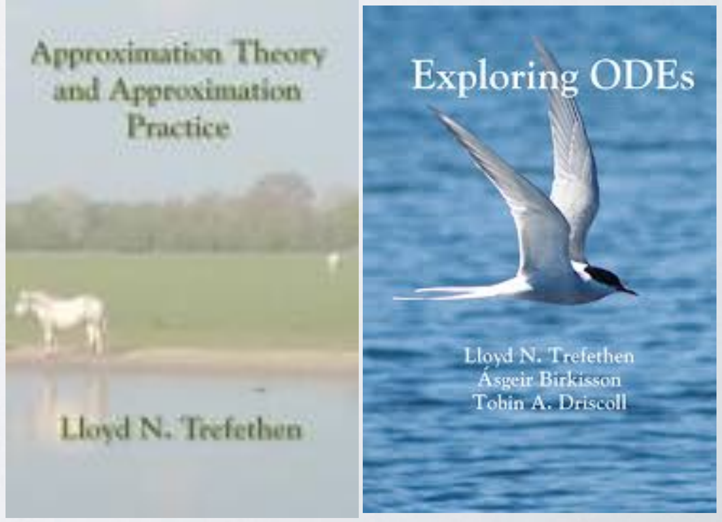
Toby Driscoll

Linear and nonlinear systems of IVPs:

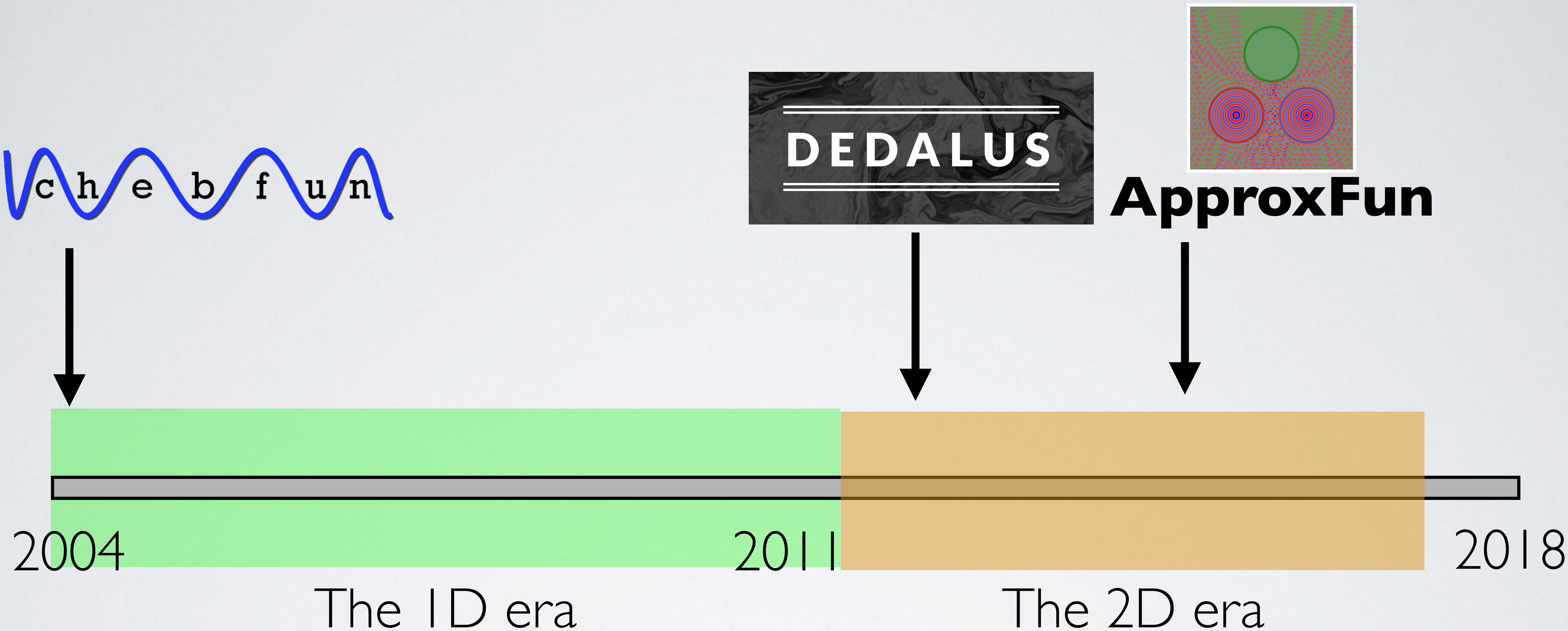
```
fun = @(t,u) [10*(u(2)-u(1)) ; ...
              28*u(1)-u(2)-u(1)*u(3) ; ...
              u(1)*u(2)-(8/3)*u(3)];
u = chebfun.ode113(fun, [0,5], [-14 -15 20]);
```



Lorenz

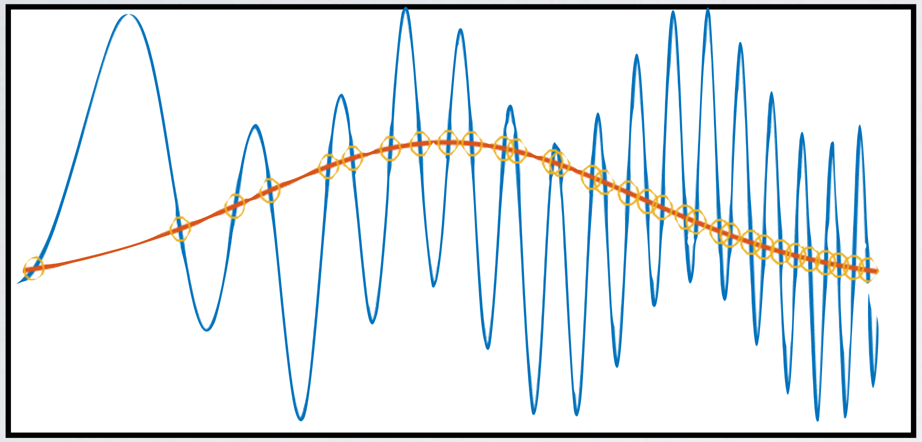


Timeline

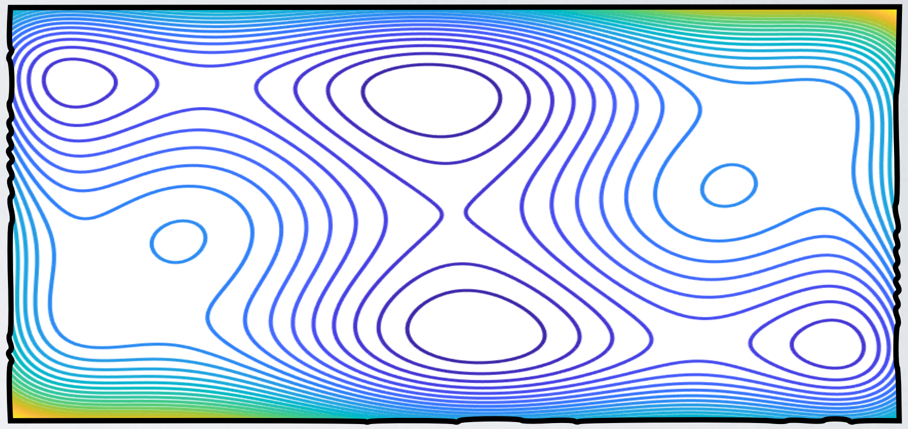


What happened between 2012-2017?
I will partly summarize five years with five ideas.

Idea 1: Leverage 1D technology for code maintenance



?



Low rank approximation

$$A \approx u_1 v_1^T + \cdots + u_r v_r^T$$

$$f(x,y) \approx g_1(y)h_1(x) + \cdots + g_r(y)h_r(x)$$

(Can also be a more efficient representation.)

Why? Integration, differentiation, evaluation, etc.

$$\int_{-1}^{1}\int_{-1}^{1} f(x,y)\,dx\,dy \approx \int_{-1}^{1} g_1(y)dy \int_{-1}^{1} h_1(x)dx + \cdots + \int_{-1}^{1} g_r(y)dy \int_{-1}^{1} h_r(x)dx$$

Computing low rank function approximants

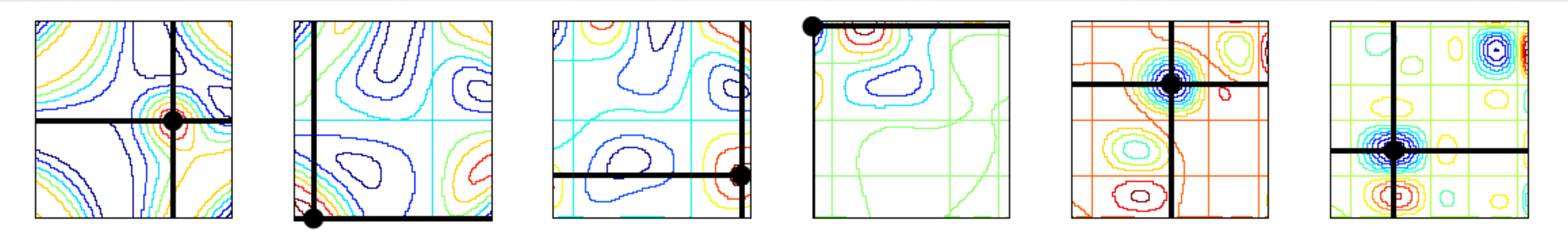
$$f(x,y) = g_1(y)h_1(x) + g_2(y)h_2(x) + \ldots$$

Ideally, SVD:

```
>> f = chebfun2( @(x,y) cos(x.*y) );
>> svd(f)
ans =
   1.896743902392399
   0.088177729243591
   0.000483326329607
   0.000001024831401
   0.000000001154335
   0.000000000000806
   0.000000000000000
```

For s.v. decay of functions:
[Smithies, 1937]
[Reade, 1981],
[Beckermann, 2004]
[Sabino, 2006]
[Beckermann & T., 2017]

How to compute it? Gaussian elimination

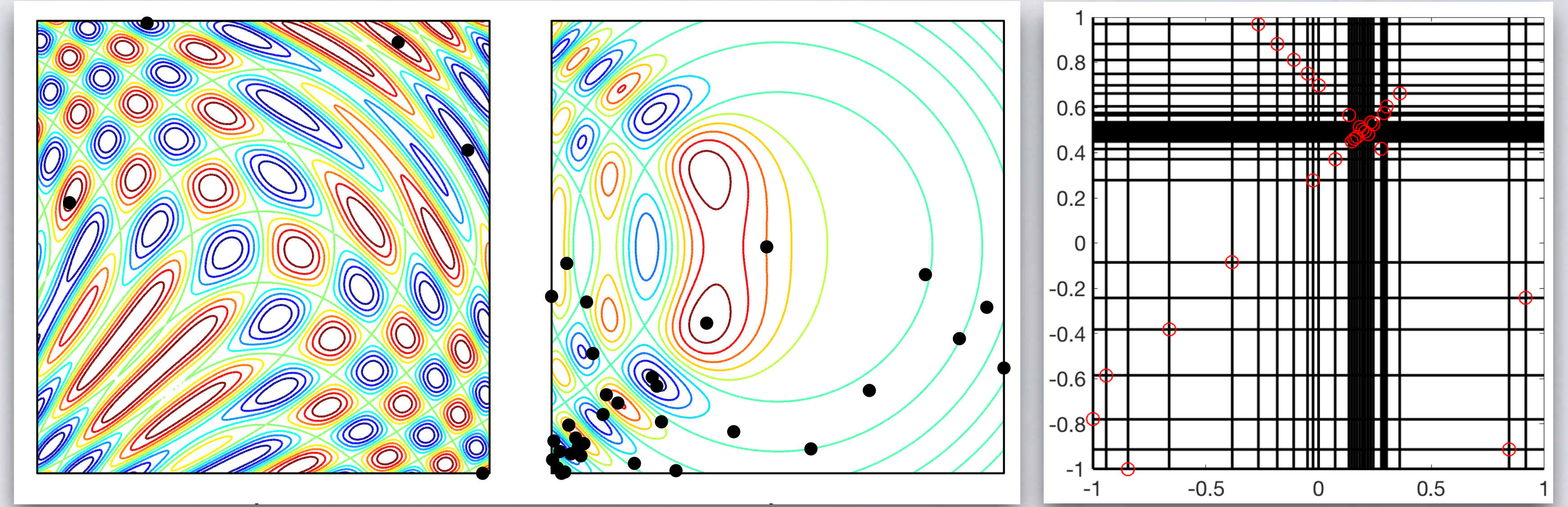


$$f \longleftarrow f - f(x_0,\cdot)f(\cdot,y_0)/f(x_0,y_0)$$

Highly related to: ACA, two-sided IDs, skeleton decomp., Geddes-Newton

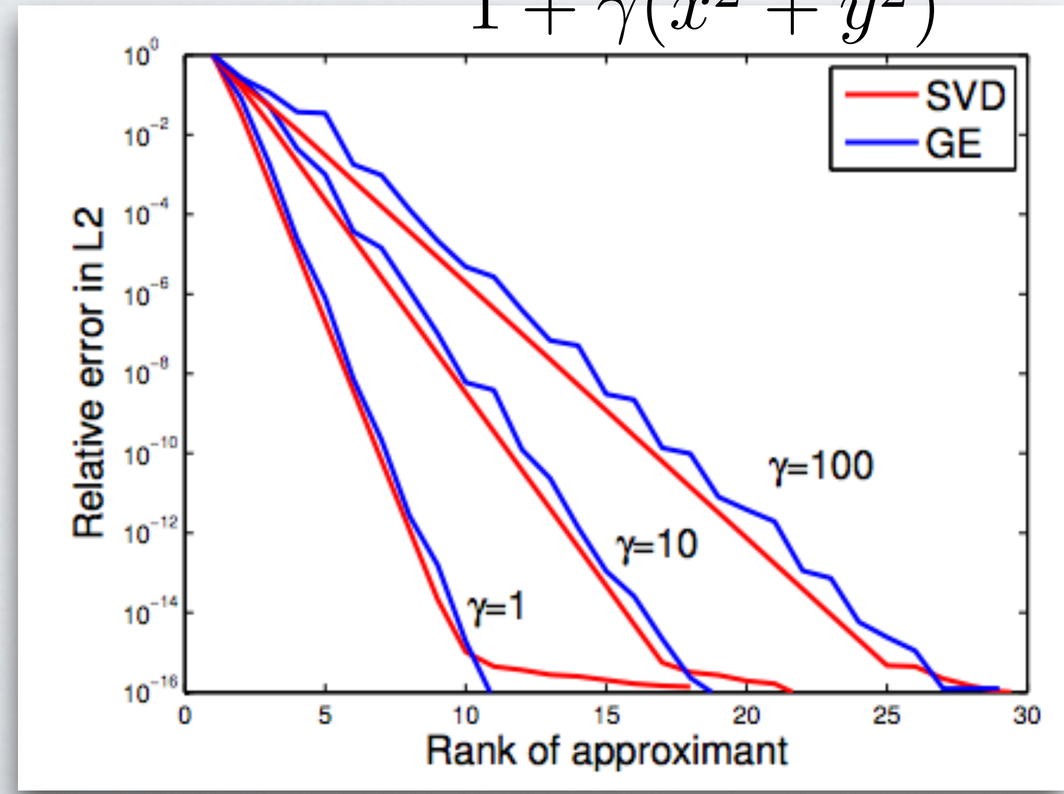
Gaussian elimination on functions

Pivot locations:



GE is a rank revealer for smooth functions:

$$f(x,y) = \frac{1}{1+\gamma(x^2+y^2)}$$



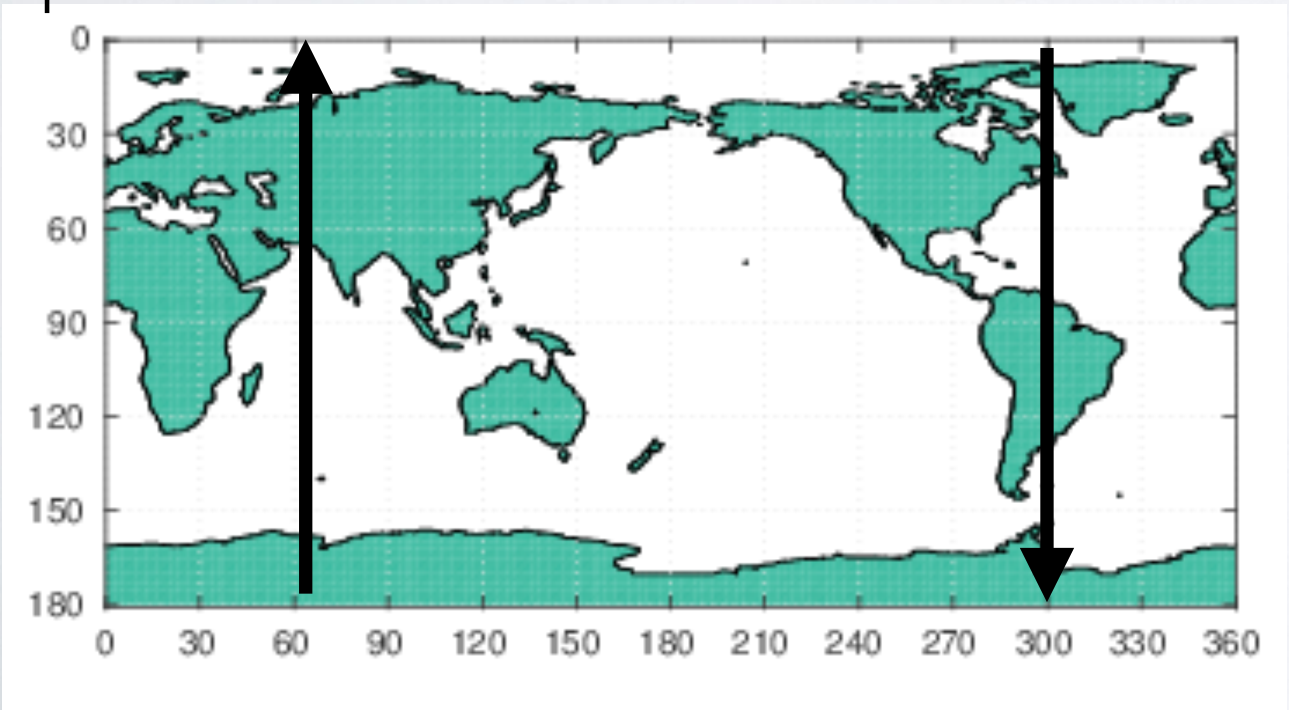
Theorem [T. & Trefethen, 2013]
If f on $[-1,1]^2$ is cont. and $f(x,\cdot)$ analytic and bounded in stadium of radius 4ρ (with $\rho > 1$). Then,

$$\left\| f - \sum_{j=1}^{r} g_j h_j^T \right\|_\infty \le C\rho^{-r}$$

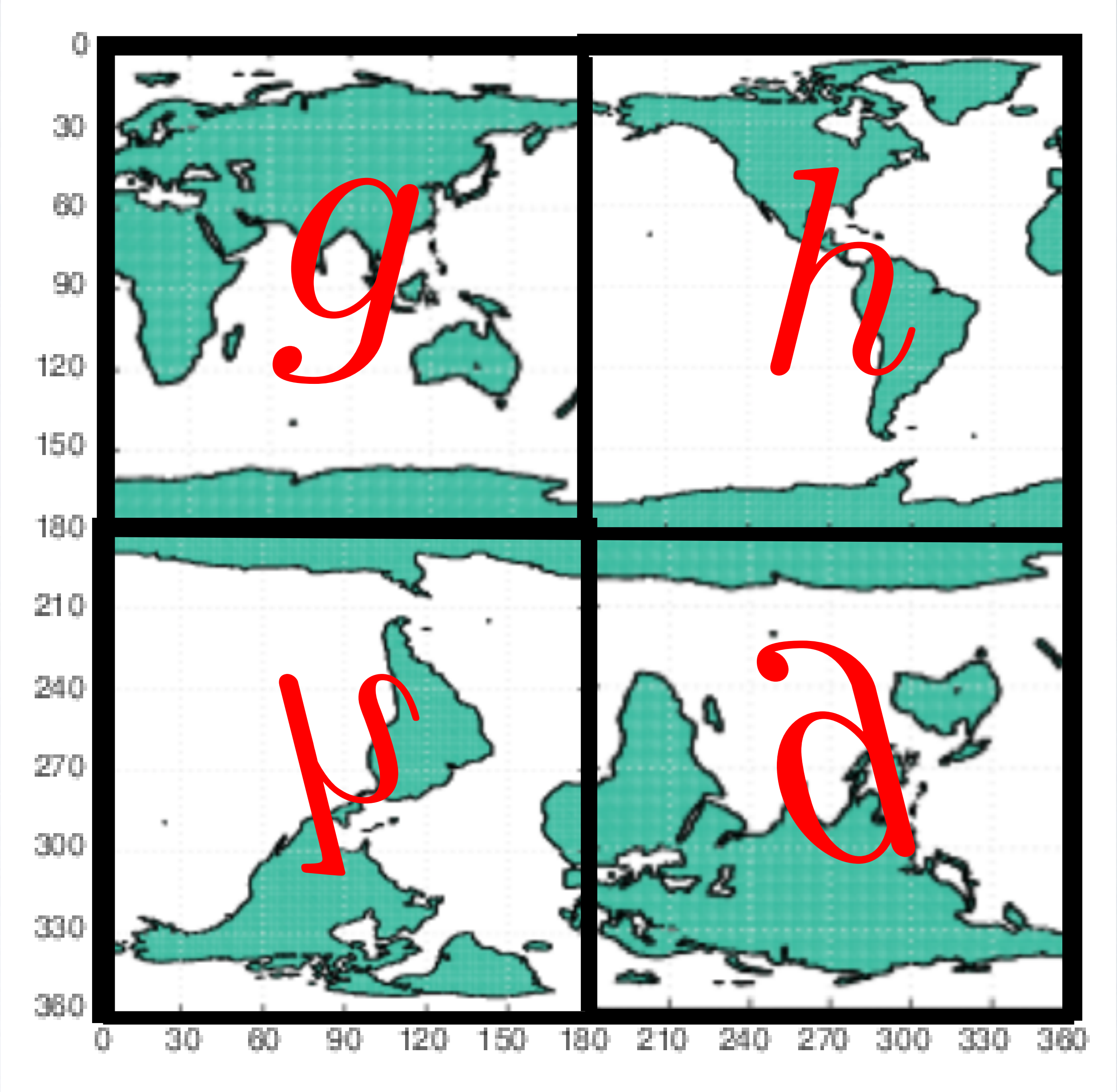
GE is robust to pivoting mistakes [T. 2016]

Idea 2: Fast transforms for highly adaptive algorithms

Spherical coordinates



Double Fourier sphere method



$$\mathcal{S}^2 \times \mathbb{Z}_2 \simeq \text{Torus}$$

[Merilees, 1973], [Orszag, 1974], [Fornberg, 1995]

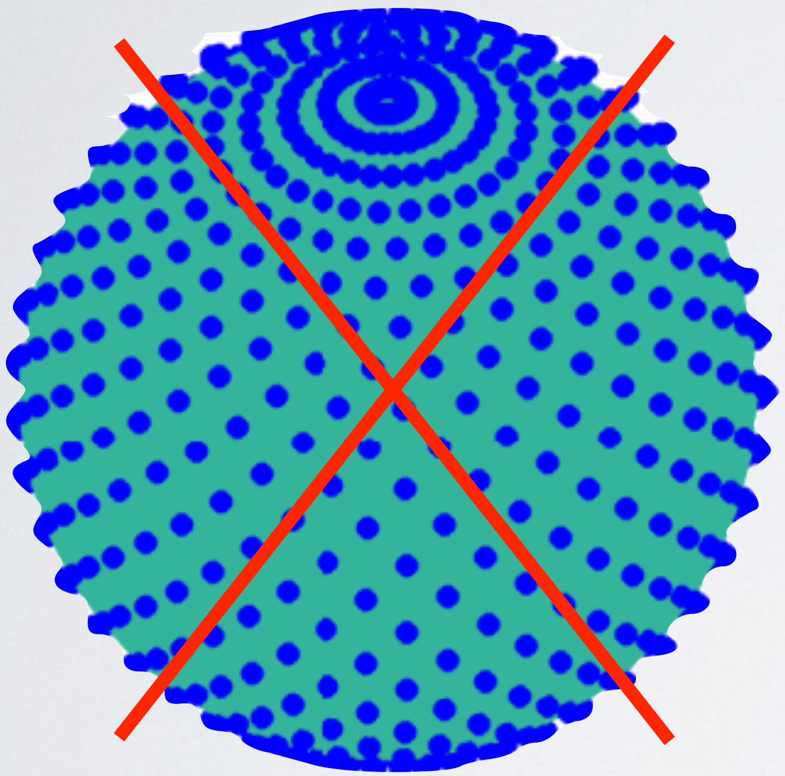
Sphere and disk

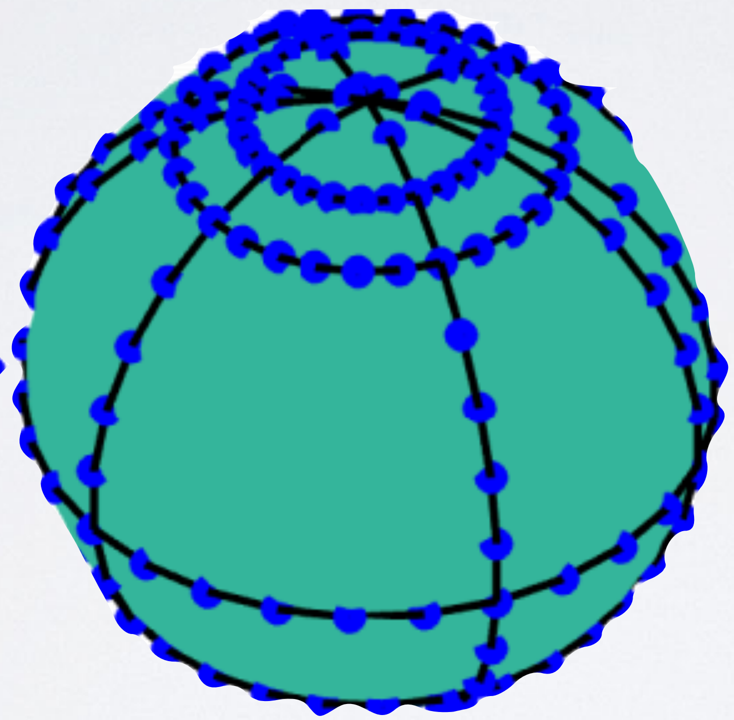
We use double Fourier sphere with structure-preserving GE.

[T., Wright, Wilber, 2016]

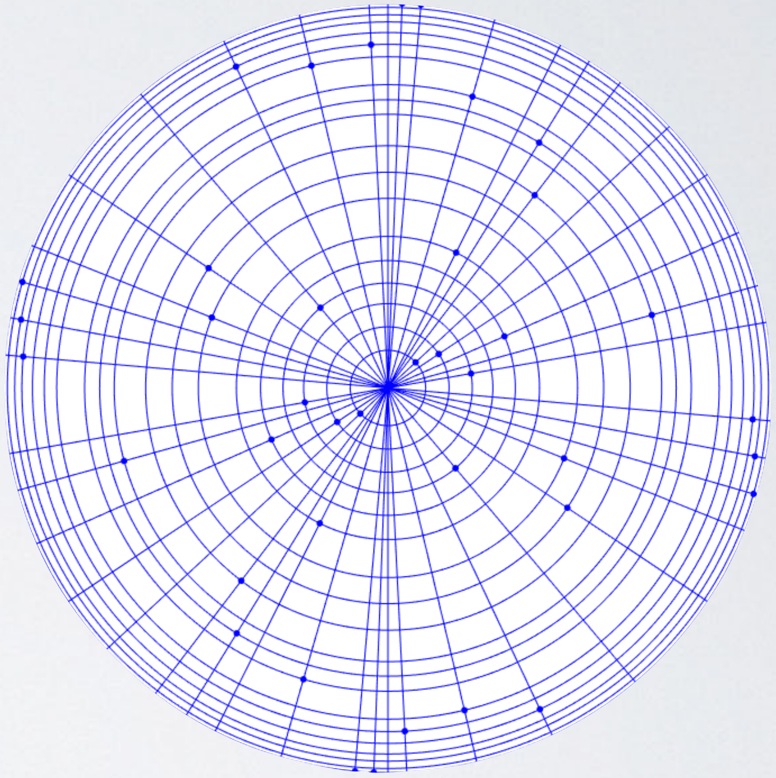
Spherefun:

```
f = spherefun(@(x,y,z) cos(5*x.*y.*z));
```





Diskfun:



Why double Fourier sphere?

We *need* fast transforms.

(While Slevinsky's SHT is fast it's not FFT speed.)

Grady Wright

Heather Wilber

Selection of FFT-based fast transforms

Discrete Fourier transform
`trigtech.vals2coeffs`, **`trigtech.coeffs2vals`**

Discrete Chebyshev transform
`coeffs2vals`, **`vals2coeffs`**

Chebyshev-to-Legendre transform
`cheb2leg`, **`leg2cheb`**

Discrete Legendre transform
`dlt`, **`idlt`**
`chebcoeffs2legvals`

Nonuniform FFTs [Ruiz & T., 2018]
`nufft`, **`inufft`**


Nick Hale


Mikael Slevinsky


Marcus Webb


Diego Ruiz

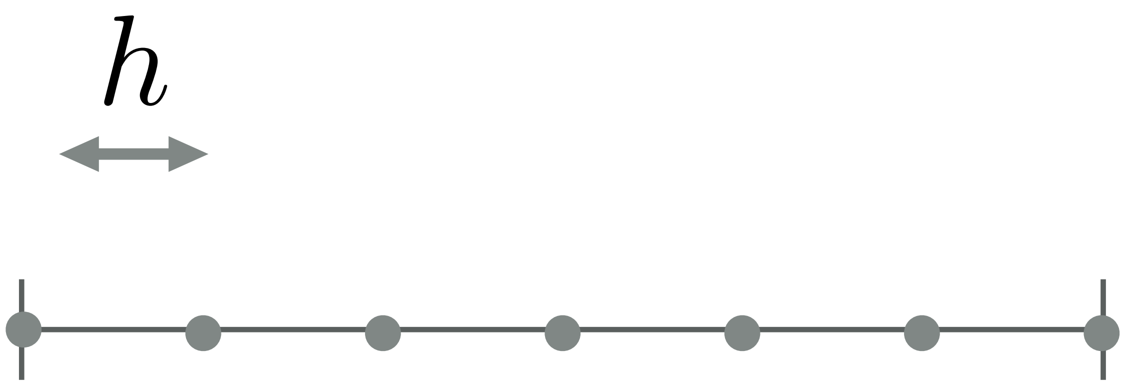
All based on FFTW so tunes to individual hardware.

Other approaches: oversampling and conv., H-matrices, asymptotics.

Idea 3: Nonperiodic analogue of the Fourier spectral method for flexibility and speed

Second-order FD

$$u''(x) = f(x), \quad u(\pm 1) = 0$$



$$u''(x_k) \approx \frac{u_{k-1} - 2u_k + u_{k+1}}{h^2}$$

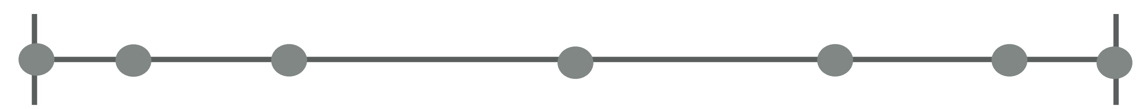
$$\begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 \\ \times & \times & \times & & & \\ & \times & \times & \times & & \\ & & \times & \times & \times & \\ & & & \times & \times & \times \\ 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} u_0 \\ u_1 \\ u_2 \\ u_3 \\ u_4 \\ u_5 \end{bmatrix} = \begin{bmatrix} 0 \\ f(x_1) \\ f(x_2) \\ f(x_3) \\ f(x_4) \\ 0 \end{bmatrix}$$

High-order limit

Spectral collocation

$$u''(x) = f(x), \quad u(\pm 1) = 0$$

Chebyshev grid



$$u(x) \approx p(x), \; u''(x_k) \approx p''(x_k)$$

$$\begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 \\ \times & \times & \times & \times & \times & \times \\ \times & \times & \times & \times & \times & \times \\ \times & \times & \times & \times & \times & \times \\ \times & \times & \times & \times & \times & \times \\ 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} u_0 \\ u_1 \\ u_2 \\ u_3 \\ u_4 \\ u_5 \end{bmatrix} = \begin{bmatrix} 0 \\ f(x_1) \\ f(x_2) \\ f(x_3) \\ f(x_4) \\ 0 \end{bmatrix}$$

[Fornberg, 1998], [Trefethen, 2000]

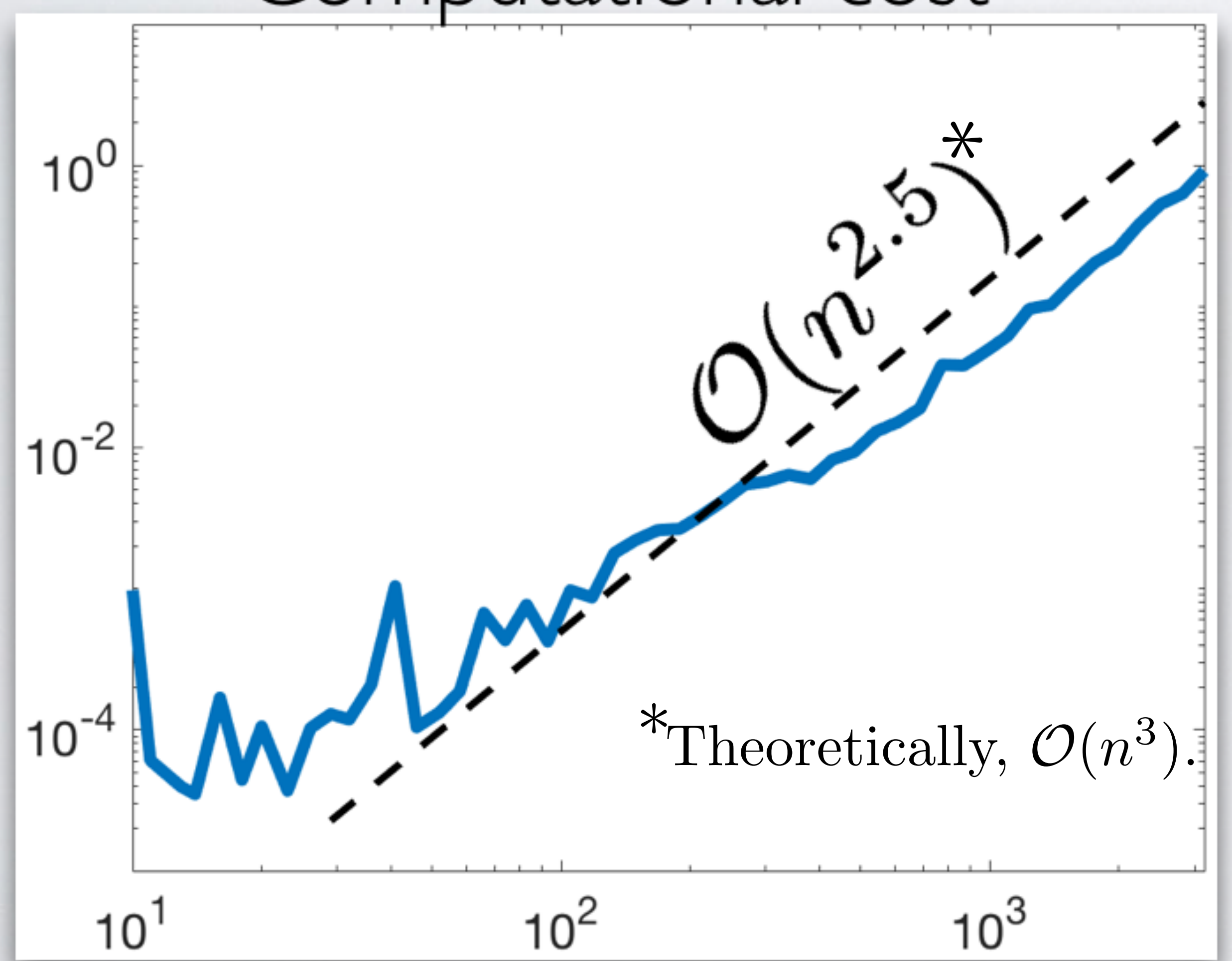
Modern version: rectangular spectral collocation [Driscoll & Hale, 2015], [Du, 2015]

Spectral collocation

"It is well-known that matrices generated by spectral methods are dense and ill-conditioned." [Chen, 2005]

Typically dense matrices

Computational cost

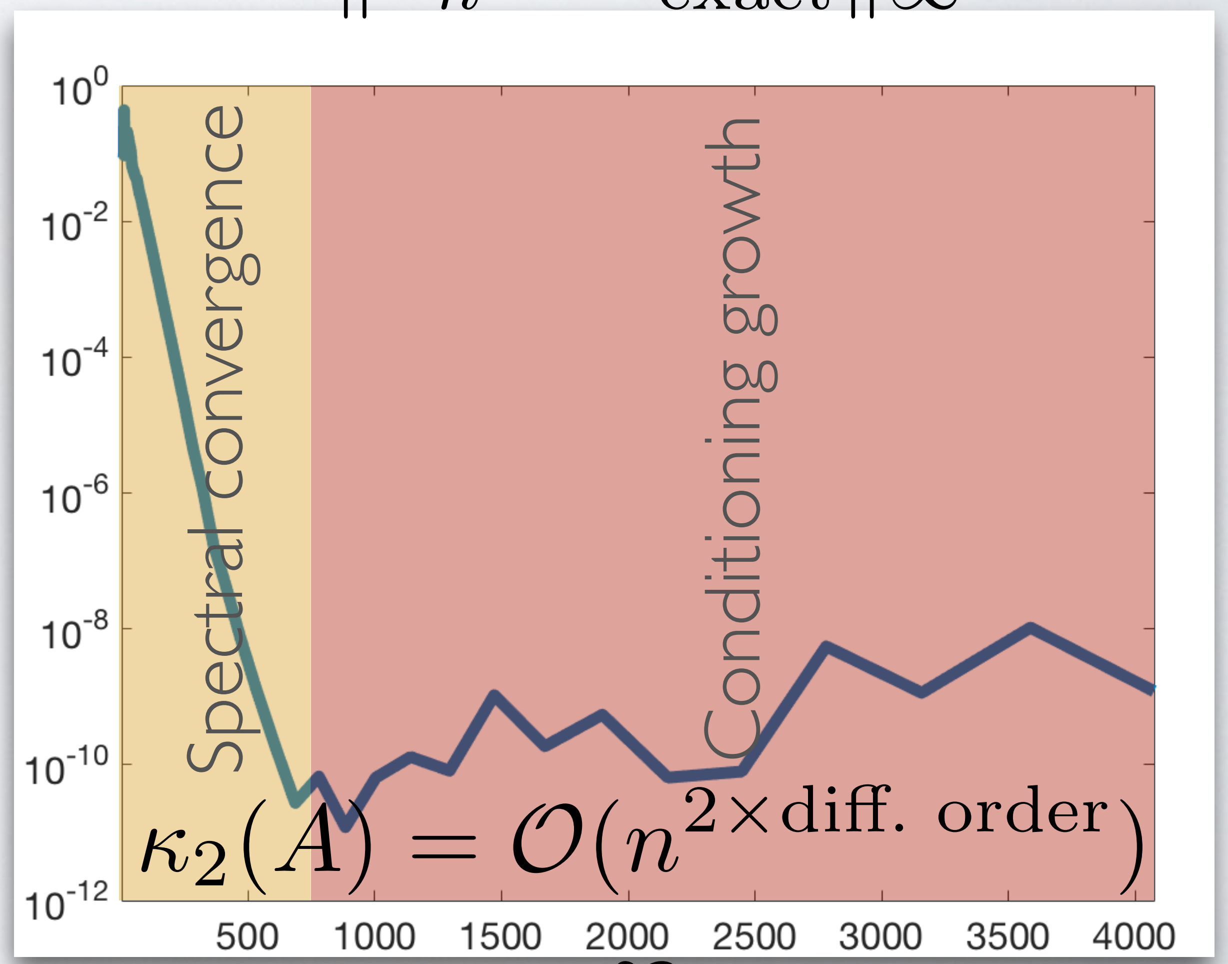


n

Off-diagonal structure [Shen, Wang, & Xia, 2016].

Typically ill-conditioned

$\|u_n - u_{\text{exact}}\|_\infty$



n

Ideas: [Du, 2015], [Wang, Samson, & Zhao, 2013]

The Fourier spectral method

Fourier spectral method

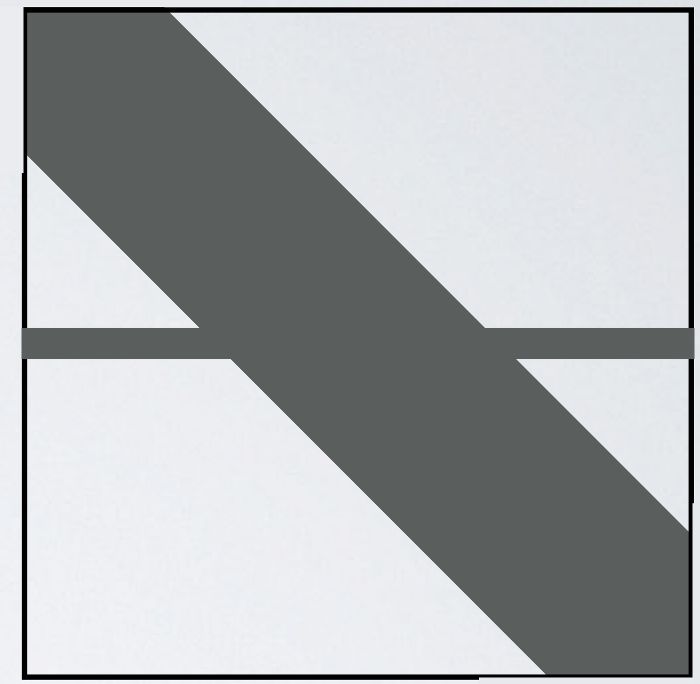
$$u''(\theta) = f(\theta), \text{ periodic}$$

$$u(\pi) = 0$$

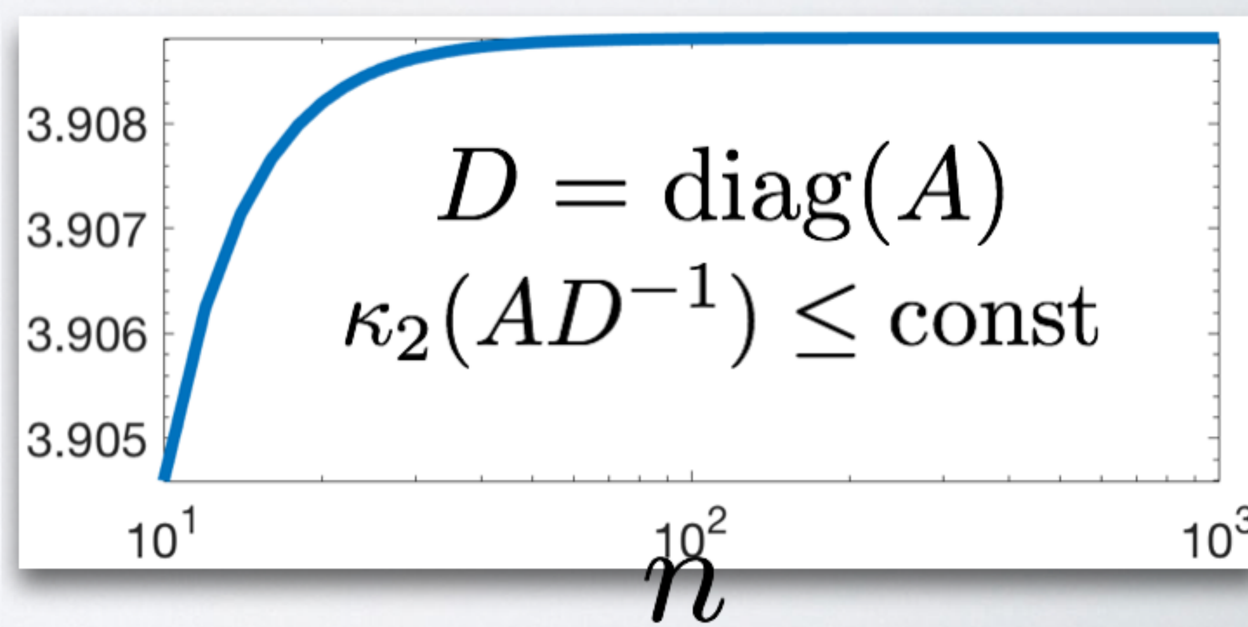
$$u(\theta) = \sum_{k=-\infty}^{\infty} u_k e^{ik\theta}$$

$$\begin{bmatrix} \ddots & & & & & & \\ & -4 & & & & & \\ & & -1 & & & & \\ \cdots & e^{-2\pi i} & e^{-\pi i} & 1 & e^{\pi i} & e^{2\pi i} & \cdots \\ & & & & -1 & & \\ & & & & & -4 & \\ & & & & & & \ddots \end{bmatrix} \begin{bmatrix} \vdots \\ u_{-2} \\ u_{-1} \\ u_0 \\ u_1 \\ u_2 \\ \vdots \end{bmatrix} = \begin{bmatrix} \vdots \\ f_{-2} \\ f_{-1} \\ 0 \\ f_1 \\ f_2 \\ \vdots \end{bmatrix}$$

Almost-banded matrices

$A =$ 

Well-conditioned matrices

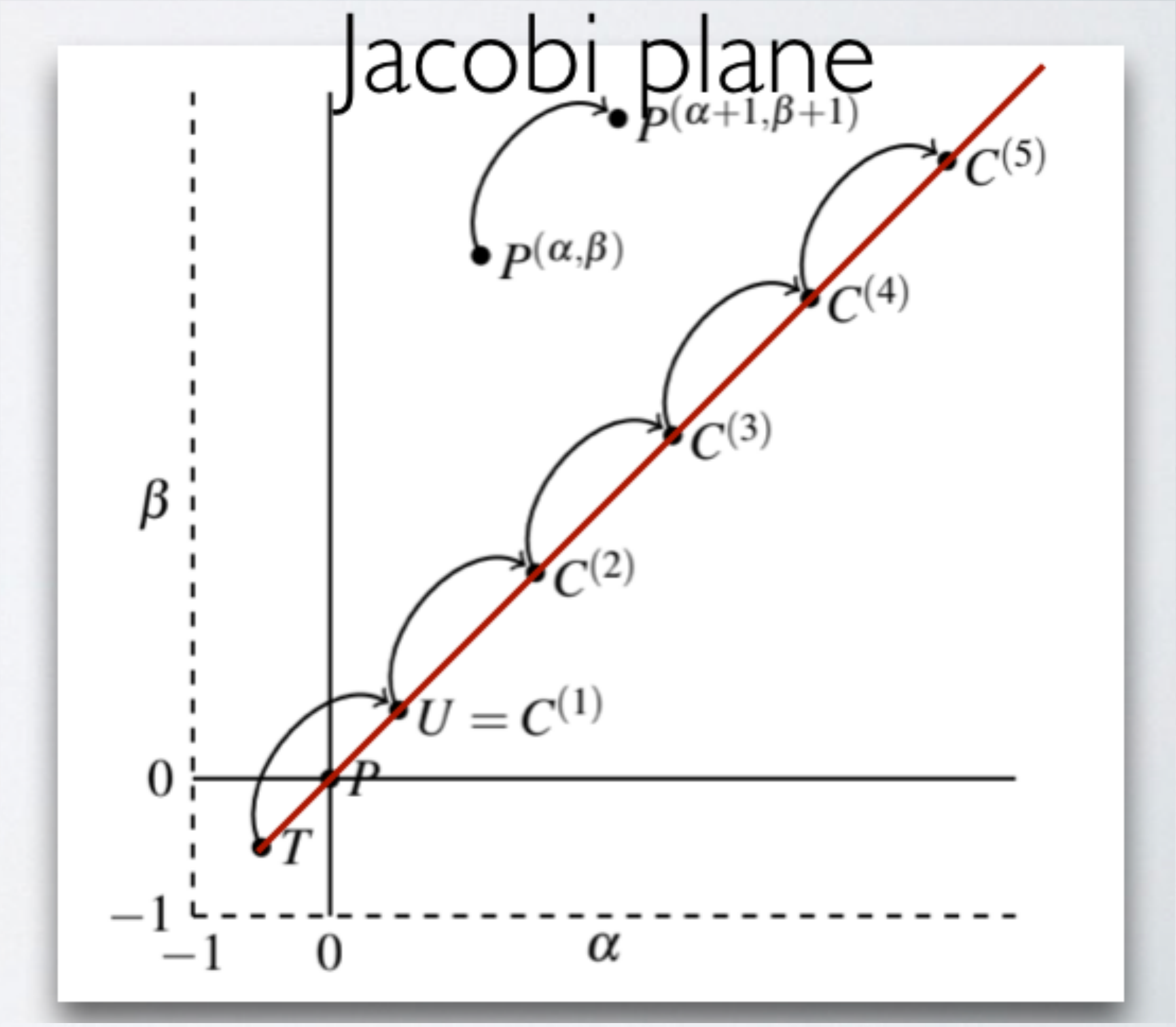


Qu: What is the non-periodic analogue?

Sparse recurrence relations

High-order derivatives:

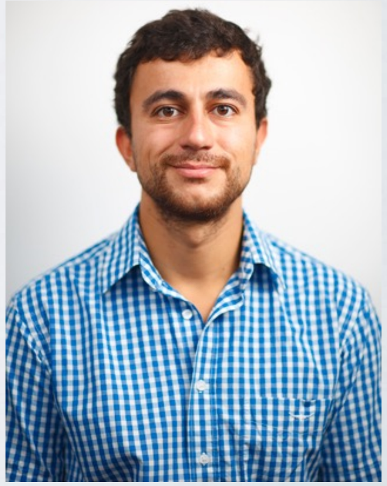
$$T_k'(x) = kU_{k-1}(x)$$
$$T_k''(x) = 2kC_{k-2}^{(2)}(x)$$
$$T_k'''(x) = 8kC_{k-3}^{(3)}(x)$$
$$T_k''''(x) = 48kC_{k-4}^{(4)}(x)$$
$$T_k'''''(x) = 384kC_{k-5}^{(5)}(x)$$



1st order recurrences:

$$T_k'(x) = kU_{k-1}(x) \quad xT_k(x) = \frac{1}{2}\left(T_{k+1}(x) + T_{k-1}(x)\right)$$
$$T_n(x) = \tfrac{1}{2}\left(U_n(x) - U_{n-2}(x)\right)$$

[DMLF, Chap. 18]



Sheehan Olver

[Olver & T., 2013]

The ultraspherical spectral method

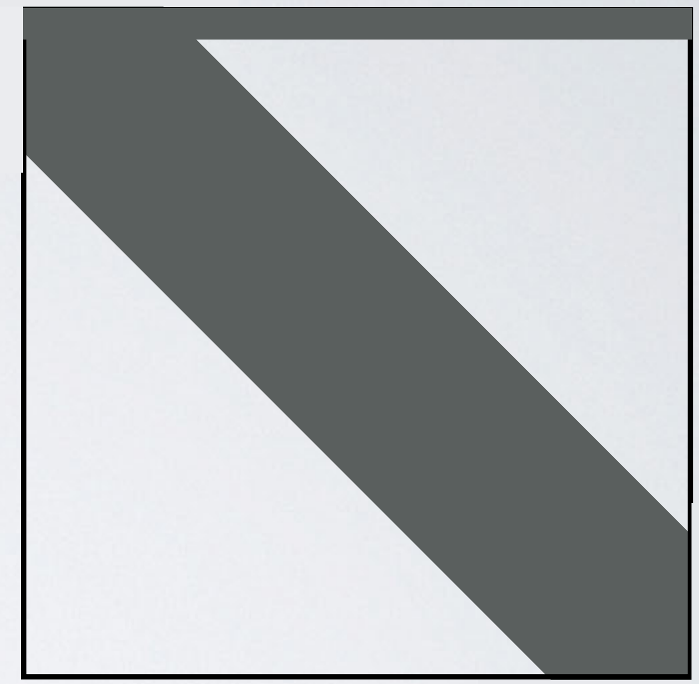
Ultraspherical method

$$\frac{du}{dx} + 4xu = 0, \qquad u(-1) = c$$

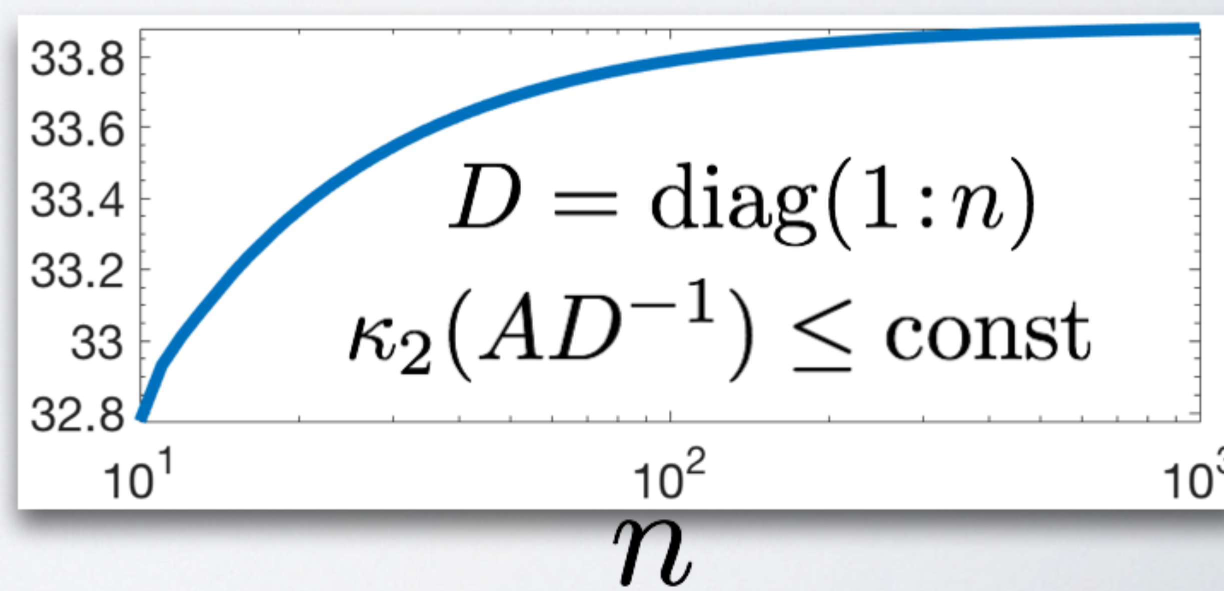
$$u(x) = \sum_{k=0}^{\infty} u_k T_k(x)$$

$$\begin{bmatrix} 1 & -1 & 1 & -1 & 1 & -1 & 1 & \cdots \\ 0 & 2 & 0 & -1 & & & & \\ 2 & & 2 & & -1 & & & \\ & 1 & & 3 & & -1 & & \\ & & 1 & & 4 & & -1 & \\ & & & 1 & & \ddots & & \ddots \\ & & & & \ddots & & & \end{bmatrix} \begin{bmatrix} u_0 \\ u_1 \\ u_2 \\ u_3 \\ u_4 \\ \vdots \\ \vdots \end{bmatrix} = \begin{bmatrix} c \\ 0 \\ 0 \\ 0 \\ 0 \\ \vdots \\ \vdots \end{bmatrix}$$

Almost-banded matrices

$A =$ 

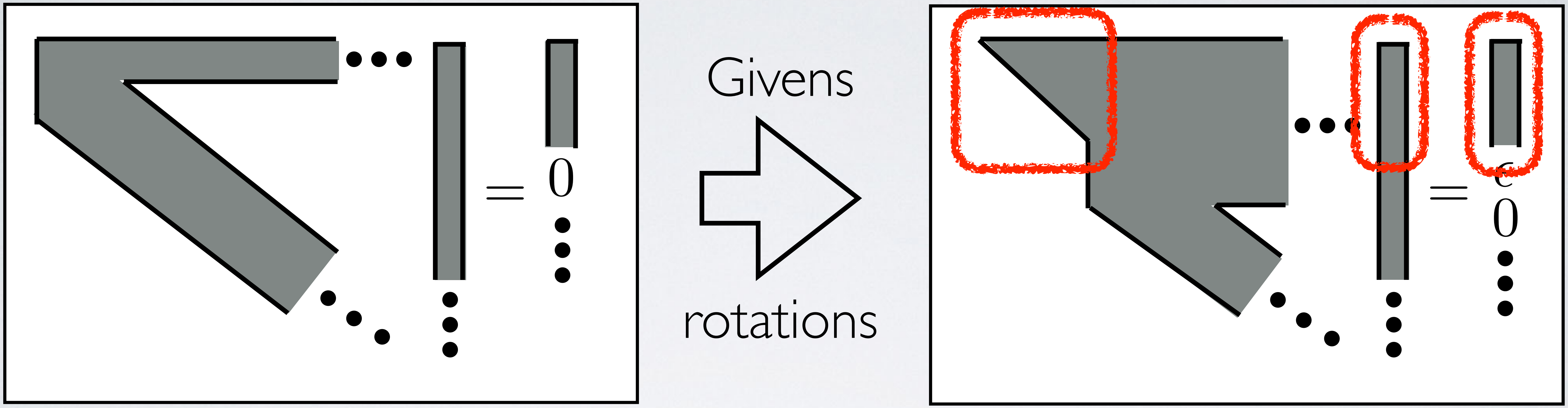
Well-conditioned matrices



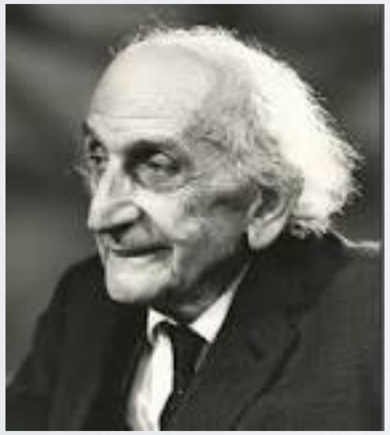
Highly related to: Petrov-Galerkin, Integral Reformulation, Integral preconditioning, Clenshaw's method, Tuckermann's lin. alg., etc.

Idea 4: Infinite-dimensional algebra for robustness

Adaptive QR



What will the backward error be? The norm of the tail: ϵ



Lanczos tau method
[Lanczos, 1938]

+

F. W. J. Olver's algorithm
[Olver, 1967]

+

Givens' rotation
[Givens, 1958]

= adaptive QR

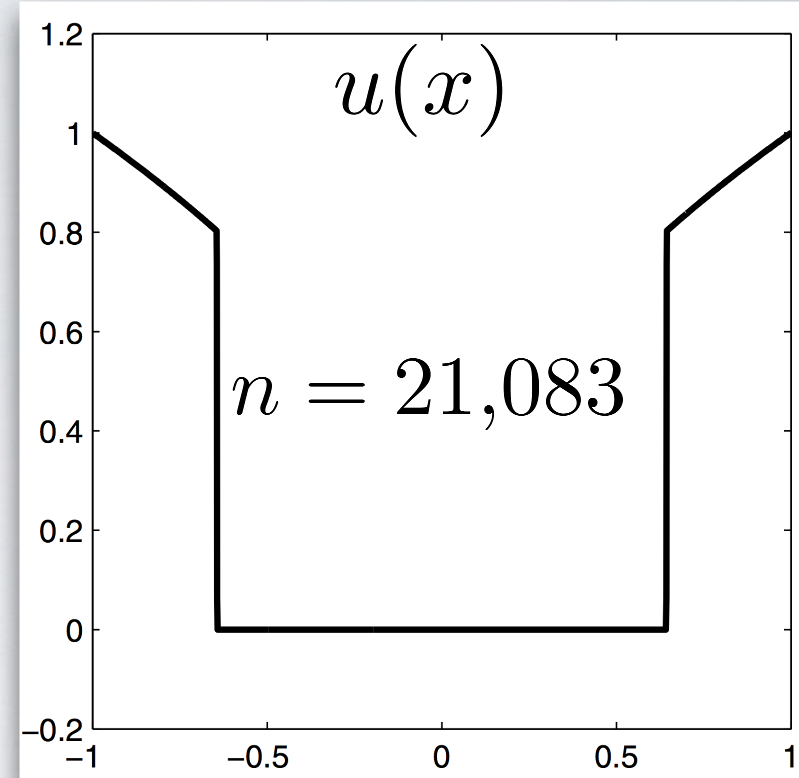
Linear complexity comes from careful data structures [Olver & T., 2014].

Two types of differential equations

1) Singularly perturbed problems

Code snippet (“Bucket equation”)

```
N = ultraop(@(x,u) 1e-7*diff(u,2) - ...
                   2*x*(cos(x)-.8)*diff(u,1) + ...
                   (cos(x)-.8)*u);
N.lbc = 1; N.rbc = 1;
tic, u = N \ 0; toc
Elapsed time is 0.033404 seconds.
```



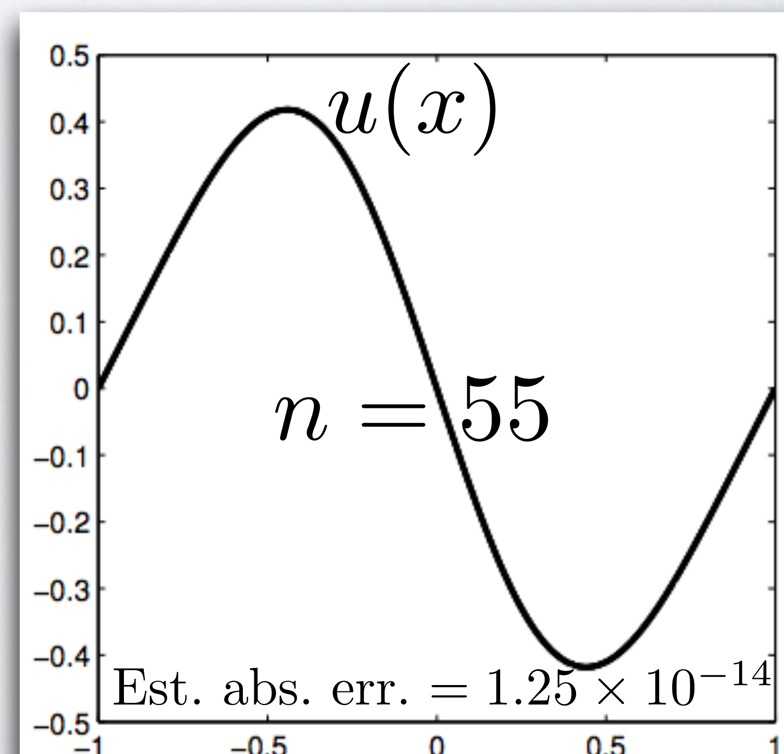
Also, adaptive subdivision: [Lee & Greengard, 1997].

2) High-order ODEs

Code snippet (“10th order ODE”)

```
x = Fun()
D = Derivative()
L=D^10 + cosh(x)*D^8 + x^3*D^6 + x^4*D^4 + cos(x)*D^2 + x^2
d = Interval()
B = [Dirichlet(d) ;
     Neumann(d)-1 ;
     [Evaluation(Interval(),first,k) for k=2:4]... ;
     [Evaluation(Interval(),last,k) for k=2:4]...]
u = [B; L] \ [ [0.,0.], [1.,1.], zeros(6)..., exp(x)]
```

ApproxFun



Active fluids: automatic code generation

Generalized Navier-Stokes equations: [Słomka & Dunkel, 2015]

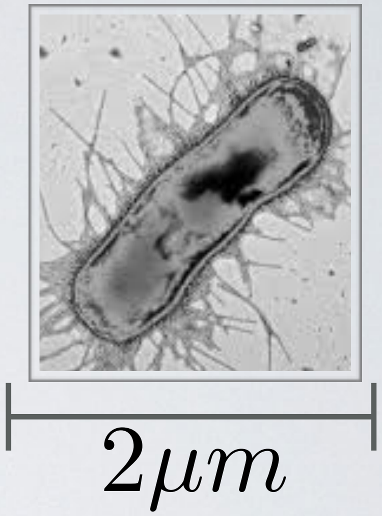
$\nabla \cdot v = 0$ Navier-Stokes activity damping

$$\partial_t v + (v \cdot \nabla) v = -\nabla p + \Gamma_0 \nabla^2 v + \Gamma_2 \nabla^4 v + \Gamma_4 \nabla^6 v$$

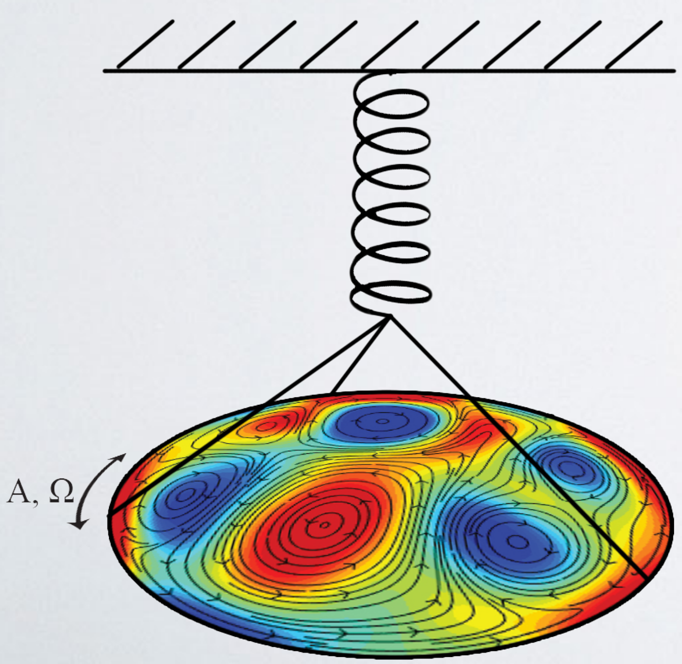
$v =$ velocity field, $p =$ internal pressure, $\Gamma_0, \Gamma_2, \Gamma_4 > 0$

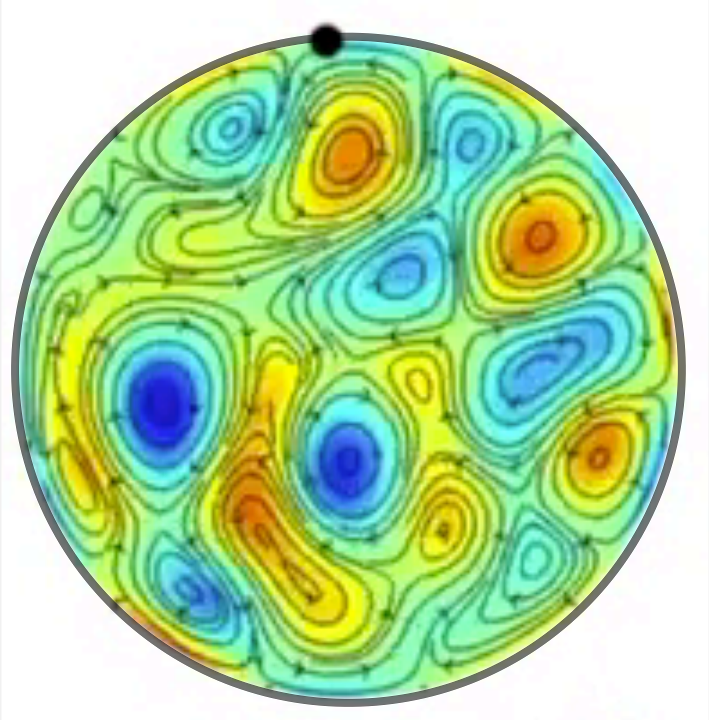
No-slip and h.o. bcs

E. Coli

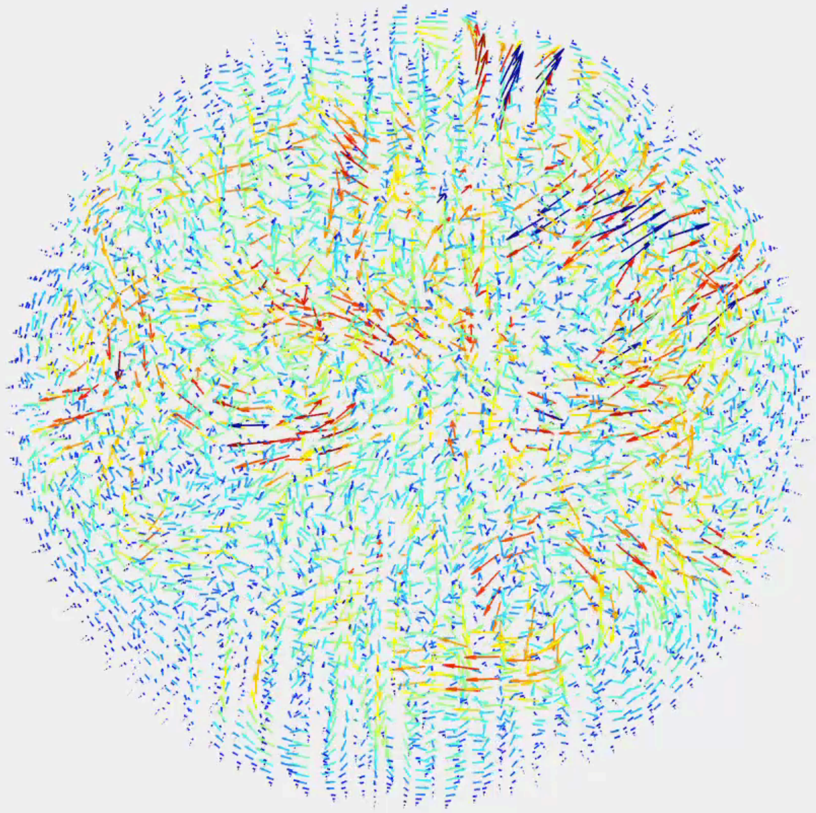


Disk, vorticity





Solid ball, velocity field



Idea 5: Exploratory PDE solvers

Separable representation

$$\mathcal{L} \approx \mathcal{L}_1^x \otimes \mathcal{L}_1^y + \cdots + \mathcal{L}_r^x \otimes \mathcal{L}_r^y$$

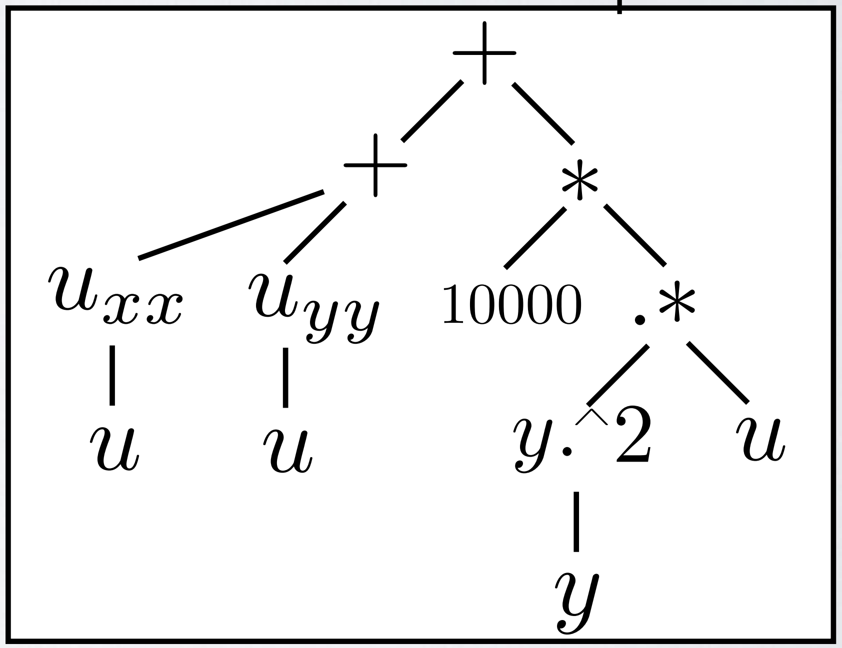
(e.g. $\nabla^2 u + \cos(xy)u$ well-approx. when $r = 7$)

Computed via a tensor-train decomposition [T. & Olver, 2014]

Chebfun2 code

```
cheb.xy
N = chebop2(@(x,y,u) lap(u) + 10000*y.^2.*u);
N.bc = 0;
u = N \ cos(x.*y);
```

AD tree for diff operator



Low-rank rep. of operator

$$\mathcal{L}u = u_{xx} + u_{yy} + 10000y^2u$$

$$\mathcal{L} = \frac{d^2}{dx^2} \otimes \mathcal{I} + \mathcal{I} \otimes \left(\frac{d^2}{dy^2} + 10000y^2\right)$$

Individually discretized by ultraspherical spectral method

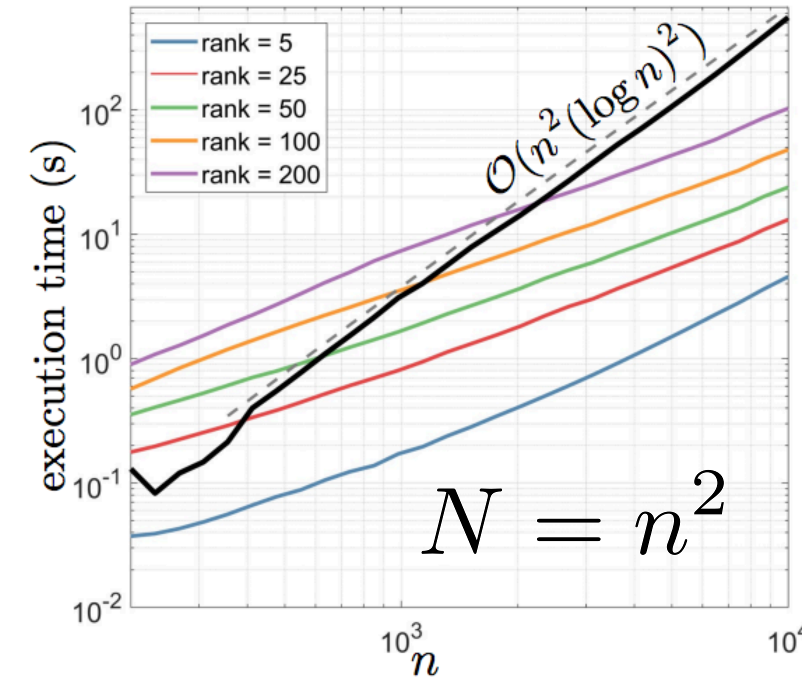
Exploratory solvers

Solve $\left(\sum_{j=1}^{r} A_j \otimes B_j\right)\mathrm{vec}(X) = \mathrm{vec}(F)$

under constraints (e.g. bcs)

$r = 2$ → $A_1 X B_1 + A_2 X B_2 = F$ with bcs

ADI-friendly? → **FI-ADI**

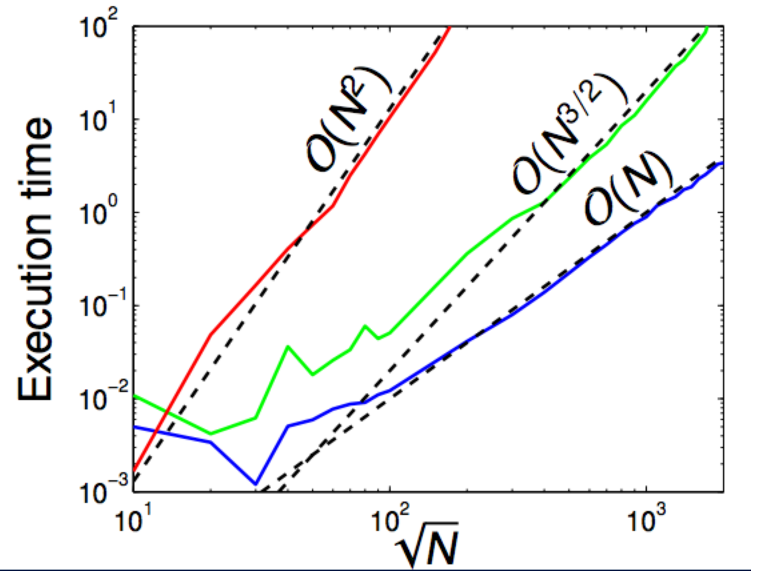


Dan Fortunato

Heather Wilber

Direct solvers:

Kronecker
Bartels-Stewart
Lyapunov



$r \neq 2$ → Kron

In the future: damped Newton, automatic IMEX splitting

Software for solving PDEs on simple geometries

<u>Simple geometries</u>

Rectangle

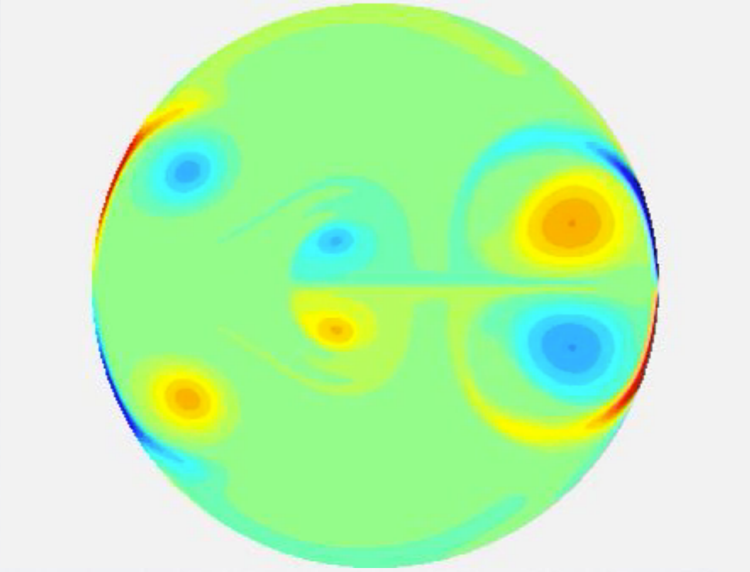
Disk

Sphere

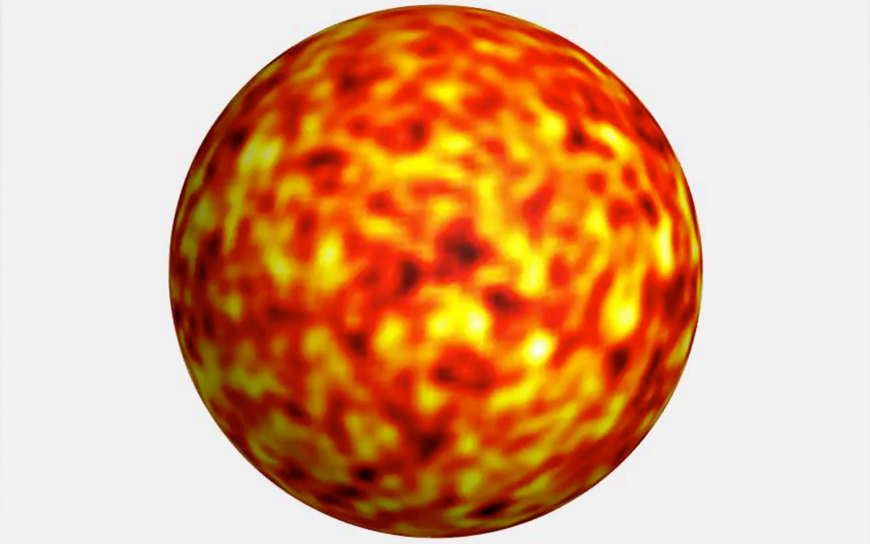
Cylinder

Solid ball

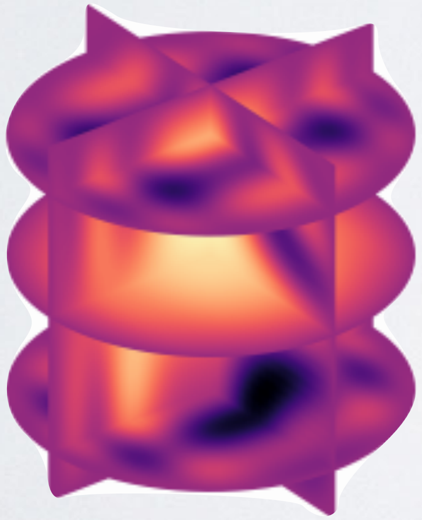
Navier-Stokes



Reaction-diffusion (Turing patterns)

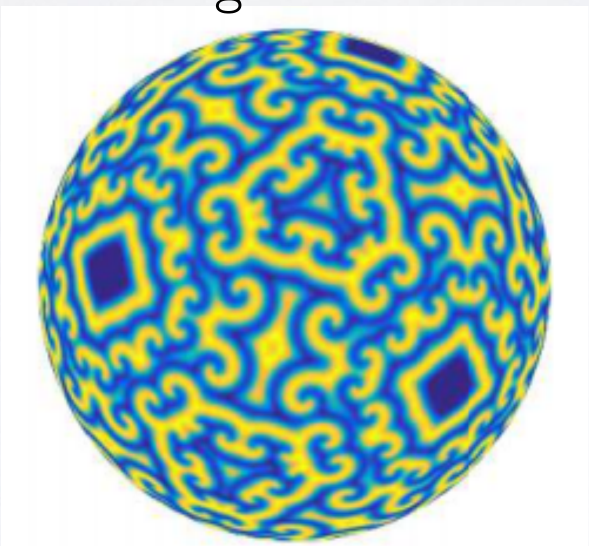


Poisson



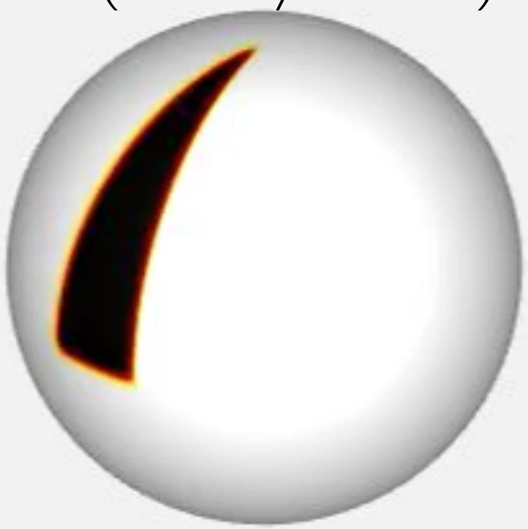
by Dan Fortunato

Ginzberg-Landau



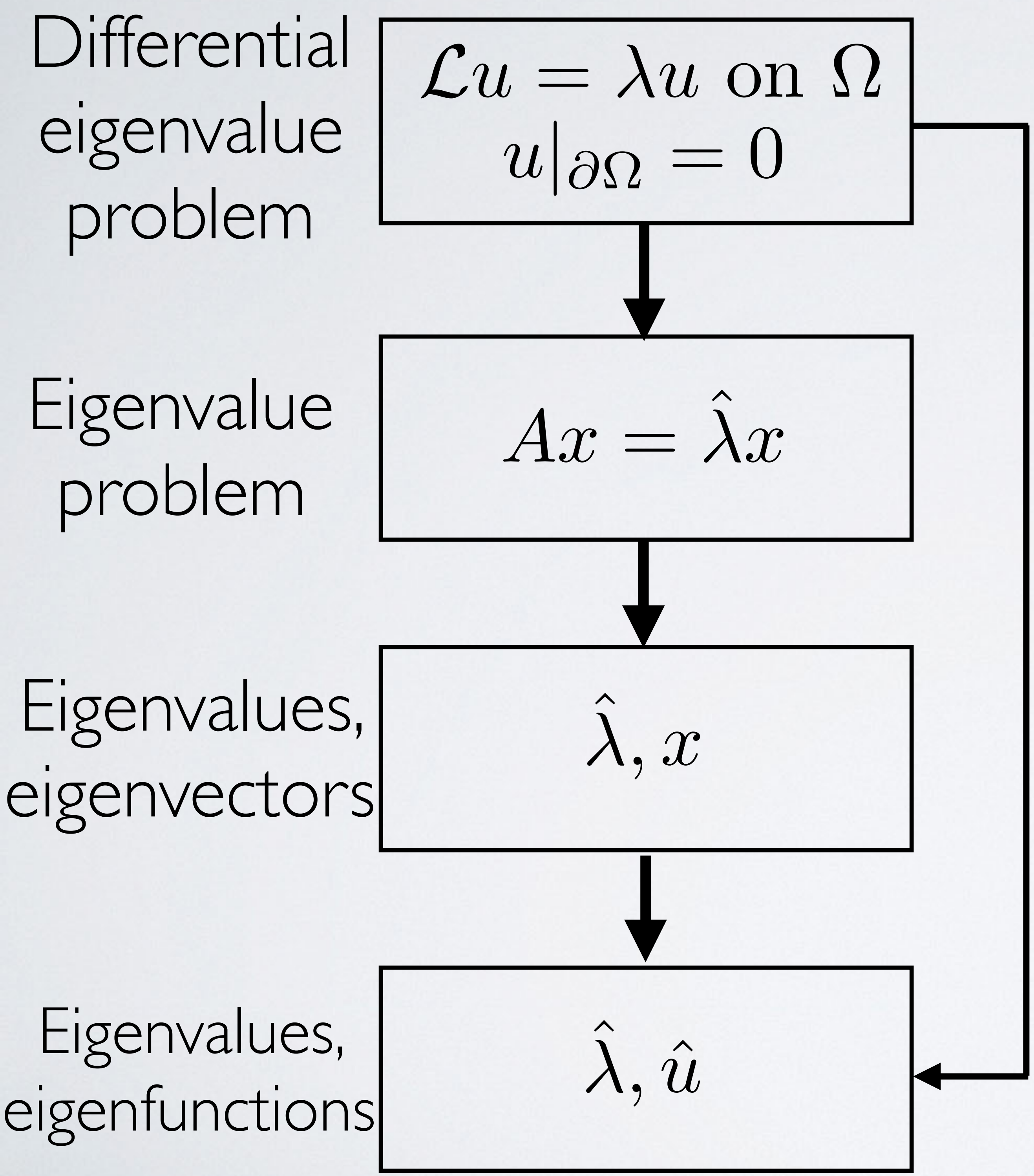
by Hadrien Montanelli

Spiral waves (Barkley model)

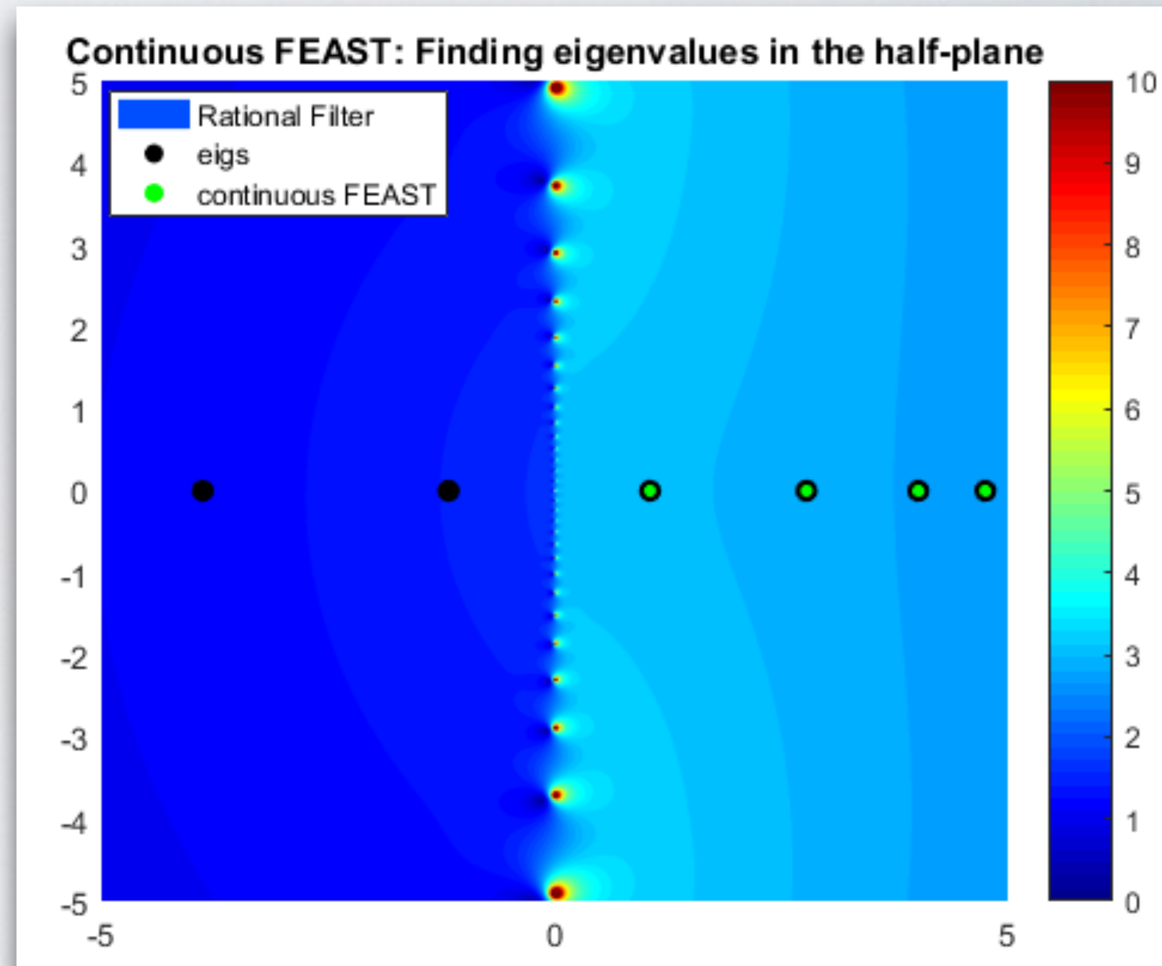


by Grady Wright

Bonus idea: Discretization oblivious algorithms



- Discretizing can increase the sensitivity of eigenvalues.



Anthony Austin

Andrew Horning

Thank you

Advertisement:

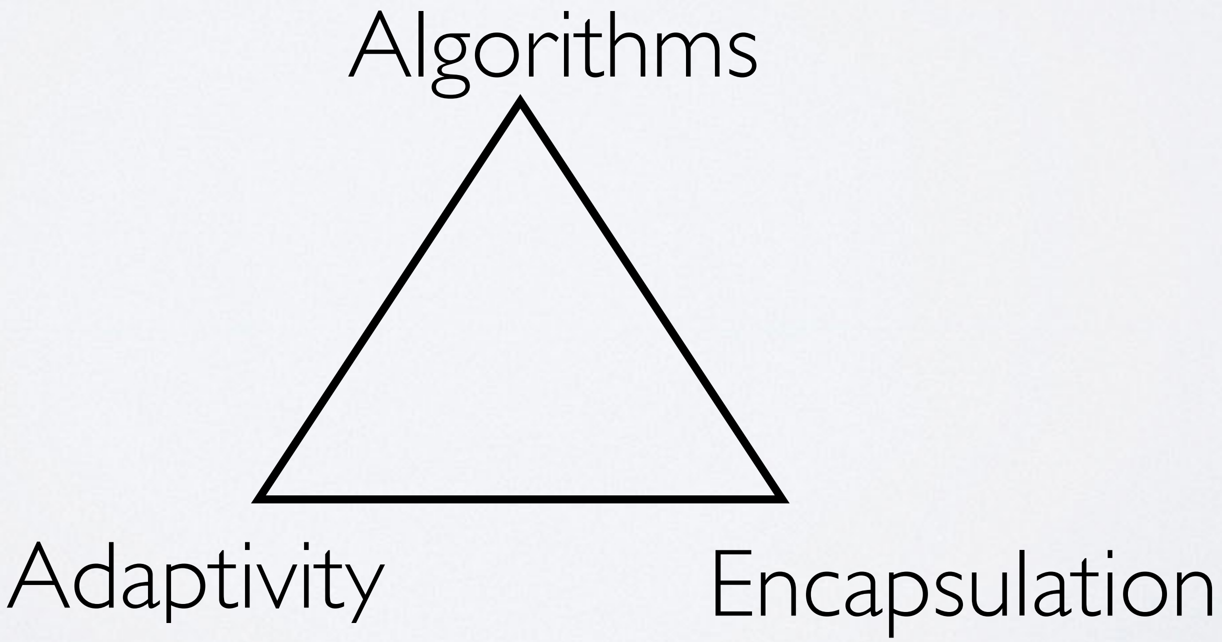
"A continuous analogoue of FEAST for differential eigenvalue problems''.

9:00-11:30am, Room 750

Andrew Horning

What if all you had to do to solve an ODE were just to write it down?

Opening line of "Exploring ODEs" [Trefethen, Driscoll, & Birkisson, 2018]



Thanks to