Fast solution of stiff PDEs in 1D, 2D and 3D

Numerical Analysis Group, Mathematical Institute

Hadrien Montanelli and Niall Bootland

January 26, 2016

UNIVERSITY OF OXFORD

Stiff PDEs in 1D, 2D and 3D

Stiff PDEs in 1D, 2D and 3D

- **Problem:** Stiff PDEs of the form

$$u_t(t, X) = \mathcal{L}u + \mathcal{N}(u), \quad t \in [0, T], \quad X \in \Omega \subset \mathbb{R}^d \ (d = 1, 2, 3),$$

with periodic boundary conditions and initial condition $u(0, X)$

Stiff PDEs in 1D, 2D and 3D

- **Problem:** Stiff PDEs of the form

$$u_t(t, X) = \mathcal{L}u + \mathcal{N}(u), \quad t \in [0, T], \quad X \in \Omega \subset \mathbb{R}^d \ (d = 1, 2, 3),$$

  with periodic boundary conditions and initial condition $u(0, X)$

- $\mathcal{L}$ is a linear differential operator of high order, e.g., $\mathcal{L}u = -u_{xxx}$ for KdV in 1D

Stiff PDEs in 1D, 2D and 3D

- **Problem:** Stiff PDEs of the form

$$u_t(t,X) = \mathcal{L}u + \mathcal{N}(u), \quad t \in [0,T], \quad X \in \Omega \subset \mathbb{R}^d \ (d = 1,2,3),$$

with periodic boundary conditions and initial condition $u(0,X)$

- $\mathcal{L}$ is a linear differential operator of high order, e.g., $\mathcal{L}u = -u_{xxx}$ for KdV in 1D
- $\mathcal{N}$ is a nonlinear (differential) operator (of lower order), e.g., $\mathcal{N}(u) = -uu_x$

Stiff PDEs in 1D, 2D and 3D

- **Problem:** Stiff PDEs of the form

$$u_t(t,X) = \mathcal{L}u + \mathcal{N}(u), \quad t \in [0,T], \quad X \in \Omega \subset \mathbb{R}^d \ (d = 1,2,3),$$

  with periodic boundary conditions and initial condition $u(0,X)$

  - $\mathcal{L}$ is a linear differential operator of high order, e.g., $\mathcal{L}u = -u_{xxx}$ for KdV in 1D
  - $\mathcal{N}$ is a nonlinear (differential) operator (of lower order), e.g., $\mathcal{N}(u) = -uu_x$

- **Method:** Fourier spectral method in space ($N = N_x N_y N_z$ points), look for

$$u(t,X) \approx \sum_{k_x=-\frac{N_x}{2}}^{\frac{N_x}{2}}{}' \sum_{k_y=-\frac{N_y}{2}}^{\frac{N_y}{2}}{}' \sum_{k_z=-\frac{N_z}{2}}^{\frac{N_z}{2}}{}' \hat{u}_{k_x k_y k_z}(t) e^{i(k_x x + k_y y + k_z z)}$$

Stiff PDEs in 1D, 2D and 3D

- **Problem:** Stiff PDEs of the form

$$u_t(t, X) = \mathcal{L}u + \mathcal{N}(u), \quad t \in [0, T], \quad X \in \Omega \subset \mathbb{R}^d \ (d = 1, 2, 3),$$

  with periodic boundary conditions and initial condition $u(0, X)$
  - $\mathcal{L}$ is a linear differential operator of high order, e.g., $\mathcal{L}u = -u_{xxx}$ for KdV in 1D
  - $\mathcal{N}$ is a nonlinear (differential) operator (of lower order), e.g., $\mathcal{N}(u) = -uu_x$
- **Method:** Fourier spectral method in space ($N = N_x N_y N_z$ points), look for

$$u(t, X) \approx \sum_{k_x=-\frac{N_x}{2}}^{\frac{N_x}{2}}{}' \sum_{k_y=-\frac{N_y}{2}}^{\frac{N_y}{2}}{}' \sum_{k_z=-\frac{N_z}{2}}^{\frac{N_z}{2}}{}' \hat{u}_{k_x k_y k_z}(t) e^{i(k_x x + k_y y + k_z z)}$$

  - This leads to a system of N ODEs for $\hat{u}(t) = \{\hat{u}_{k_x k_y k_z}(t)\}$,

$$\hat{u}'(t) = \mathbf{L}\hat{u} + \mathbf{N}(\hat{u}), \quad t \in [0, T],$$

  with initial condition $\hat{u}(0)$

From `pde15s` to `spin`

From `pde15s` to `spin`

- **Problem:** System of ODEs of the form

$$u'(t) = \mathbf{L}u + \mathbf{N}(u), \quad t \in [0, T],$$

with initial condition $u(0)$

From `pde15s` to `spin`

- **Problem:** System of ODEs of the form

$$u'(t) = \mathbf{L}u + \mathbf{N}(u), \quad t \in [0, T],$$

  with initial condition $u(0)$

- Chebfun 1D code `pde15s` uses MATLAB `ode15s` command

From `pde15s` to `spin`

- **Problem:** System of ODEs of the form

$$u'(t) = \mathbf{L}u + \mathbf{N}(u), \quad t \in [0, T],$$

  with initial condition $u(0)$

- Chebfun 1D code `pde15s` uses MATLAB `ode15s` command
- Large eigenvalues of $L \Rightarrow$ explicit methods need very small time-steps (stiffness)

From `pde15s` to `spin`

- **Problem:** System of ODEs of the form

$$u'(t) = \mathbf{L}u + \mathbf{N}(u), \quad t \in [0, T],$$

  with initial condition $u(0)$
  - Chebfun 1D code `pde15s` uses MATLAB `ode15s` command
  - Large eigenvalues of $L \Rightarrow$ explicit methods need very small time-steps (stiffness)
- **Method:** Time-stepping with exponential integrators

From `pde15s` to `spin`

- **Problem:** System of ODEs of the form

$$u'(t) = \mathbf{L}u + \mathbf{N}(u), \quad t \in [0, T],$$

  with initial condition $u(0)$
  - Chebfun 1D code `pde15s` uses MATLAB `ode15s` command
  - Large eigenvalues of $L \Rightarrow$ explicit methods need very small time-steps (stiffness)
- **Method:** Time-stepping with exponential integrators
  - Integrate $\mathbf{L}$ exactly and use a numerical scheme for $\mathbf{N}$

From `pde15s` to `spin`

- **Problem:** System of ODEs of the form

$$u'(t) = \mathbf{L}u + \mathbf{N}(u), \quad t \in [0, T],$$

  with initial condition $u(0)$
  - Chebfun 1D code `pde15s` uses MATLAB `ode15s` command
  - Large eigenvalues of $L \Rightarrow$ explicit methods need very small time-steps (stiffness)
- **Method:** Time-stepping with exponential integrators
  - Integrate $\mathbf{L}$ exactly and use a numerical scheme for $\mathbf{N}$
  - Numerical scheme for $\mathbf{N}$: one-step (ETD Runge-Kutta), multistep (ETD Adams-Bashforth), or a combination (Lawson and Predictor-Corrector)

From `pde15s` to `spin`

- **Problem:** System of ODEs of the form

$$u'(t) = \mathbf{L}u + \mathbf{N}(u), \quad t \in [0, T],$$

  with initial condition $u(0)$

  - Chebfun 1D code `pde15s` uses MATLAB `ode15s` command
  - Large eigenvalues of $L \Rightarrow$ explicit methods need very small time-steps (stiffness)

- **Method:** Time-stepping with exponential integrators
  - Integrate $\mathbf{L}$ exactly and use a numerical scheme for $\mathbf{N}$
  - Numerical scheme for $\mathbf{N}$: one-step (ETD Runge-Kutta), multistep (ETD Adams-Bashforth), or a combination (Lawson and Predictor-Corrector)
  - They go back to Hersch (1958) and Certaine (1960)

From `pde15s` to `spin`

- **Problem:** System of ODEs of the form

$$u'(t) = \mathbf{L}u + \mathbf{N}(u), \quad t \in [0, T],$$

  with initial condition $u(0)$
  - Chebfun 1D code `pde15s` uses MATLAB `ode15s` command
  - Large eigenvalues of $L \Rightarrow$ explicit methods need very small time-steps (stiffness)
- **Method:** Time-stepping with exponential integrators
  - Integrate $\mathbf{L}$ exactly and use a numerical scheme for $\mathbf{N}$
  - Numerical scheme for $\mathbf{N}$: one-step (ETD Runge-Kutta), multistep (ETD Adams-Bashforth), or a combination (Lawson and Predictor-Corrector)
  - They go back to Hersch (1958) and Certaine (1960)
- **Q1:** Are exponential integrators competitive stiff solvers in 1D, 2D and 3D?

From `pde15s` to `spin`

- **Problem:** System of ODEs of the form

$$u'(t) = \mathbf{L}u + \mathbf{N}(u), \quad t \in [0, T],$$

  with initial condition $u(0)$
  - Chebfun 1D code `pde15s` uses MATLAB `ode15s` command
  - Large eigenvalues of $L \Rightarrow$ explicit methods need very small time-steps (stiffness)
- **Method:** Time-stepping with exponential integrators
  - Integrate **L** exactly and use a numerical scheme for **N**
  - Numerical scheme for **N**: one-step (ETD Runge-Kutta), multistep (ETD Adams-Bashforth), or a combination (Lawson and Predictor-Corrector)
  - They go back to Hersch (1958) and Certaine (1960)
- **Q1:** Are exponential integrators competitive stiff solvers in 1D, 2D and 3D?
- **Q2:** What is the most accurate, stable and efficient exponential integrator?

From `pde15s` to `spin`

- **Problem:** System of ODEs of the form

$$u'(t) = \mathbf{L}u + \mathbf{N}(u), \quad t \in [0, T],$$

  with initial condition $u(0)$

  - Chebfun 1D code `pde15s` uses MATLAB `ode15s` command
  - Large eigenvalues of $L \Rightarrow$ explicit methods need very small time-steps (stiffness)
- **Method:** Time-stepping with exponential integrators
  - Integrate $\mathbf{L}$ exactly and use a numerical scheme for $\mathbf{N}$
  - Numerical scheme for $\mathbf{N}$: one-step (ETD Runge-Kutta), multistep (ETD Adams-Bashforth), or a combination (Lawson and Predictor-Corrector)
  - They go back to Hersch (1958) and Certaine (1960)
- **Q1:** Are exponential integrators competitive stiff solvers in 1D, 2D and 3D?
- **Q2:** What is the most accurate, stable and efficient exponential integrator?
- **Q3:** Is it possible to use adaptivity in both time and space?

Exponential integrators for $u' = \mathbf{L}u + \mathbf{N}(u)$

Exponential integrators for $u' = \mathbf{L}u + \mathbf{N}(u)$

- For given starting values $u_0, u_1, \ldots, u_{q-1}$ at times $t = 0, h, \ldots, (q-1)h$, the numerical approximation u_{n+1} at time $t_{n+1} = (n+1)h$, $n+1 \geq q$, is given by

$$u_{n+1} = e^{hL}u_n + h\sum_{i=1}^{s} B_i(hL)\mathbf{N}(v_i) + h\sum_{i=1}^{q-1} V_i(hL)\mathbf{N}(u_{n-i}),$$

with internal stages $v_1 = u_n$ and, for $2 \leq i \leq s$,

$$v_i = e^{C_i hL}u_n + h\sum_{j=1}^{i-1} A_{i,j}(hL)\mathbf{N}(v_j) + h\sum_{j=1}^{q-1} U_{i,j}(hL)\mathbf{N}(u_{n-j})$$

Exponential integrators for $u' = \mathbf{L}u + \mathbf{N}(u)$

- For given starting values $u_0, u_1, \dots, u_{q-1}$ at times $t = 0, h, \dots, (q-1)h$, the numerical approximation u_{n+1} at time $t_{n+1} = (n+1)h$, $n+1 \geq q$, is given by

$$u_{n+1} = e^{hL} u_n + h \sum_{i=1}^{s} B_i(hL)\mathbf{N}(v_i) + h \sum_{i=1}^{q-1} V_i(hL)\mathbf{N}(u_{n-i}),$$

with internal stages $v_1 = u_n$ and, for $2 \leq i \leq s$,

$$v_i = e^{C_i hL} u_n + h \sum_{j=1}^{i-1} A_{i,j}(hL)\mathbf{N}(v_j) + h \sum_{j=1}^{q-1} U_{i,j}(hL)\mathbf{N}(u_{n-j})$$

- The s internal stages are approximations of u at times between t_n and t_{n+1}

Exponential integrators for $u' = \mathbf{L}u + \mathbf{N}(u)$

- For given starting values $u_0, u_1, \ldots, u_{q-1}$ at times $t = 0, h, \ldots, (q-1)h$, the numerical approximation u_{n+1} at time $t_{n+1} = (n+1)h$, $n+1 \geq q$, is given by

$$u_{n+1} = e^{hL} u_n + h \sum_{i=1}^{s} B_i(hL)\mathbf{N}(v_i) + h \sum_{i=1}^{q-1} V_i(hL)\mathbf{N}(u_{n-i}),$$

  with internal stages $v_1 = u_n$ and, for $2 \leq i \leq s$,

$$v_i = e^{C_i hL} u_n + h \sum_{j=1}^{i-1} A_{i,j}(hL)\mathbf{N}(v_j) + h \sum_{j=1}^{q-1} U_{i,j}(hL)\mathbf{N}(u_{n-j})$$

- The s internal stages are approximations of u at times between t_n and t_{n+1}
- The q steps are the values of u at previous times t_n, t_{n-1}, ..., t_{n-q+1}

Comparison of exponential integrators (Q2)

Method	Type	Order	Stages s	Steps q
ABNørsett4	ETD Adams-Bashfort	4	1	4
ABNørsett5	ETD Adams-Bashfort	5	1	5
ABNørsett6	ETD Adams-Bashfort	6	1	6
Friedli (VRK4)	ETD Runge-Kutta	4	4	1
Strehmel-Weiner	ETD Runge-Kutta	4	4	1
Cox-Matthews (ETDRK4)	ETD Runge-Kutta	4	4	1
Krogstad (ETDRK4-B)	ETD Runge-Kutta	4	4	1
Minchev	ETD Runge-Kutta	4	4	1
Hochbruck-Ostermann	ETD Runge-Kutta	4	5	1
Luan-Ostermann (EXPRK5S8)	ETD Runge-Kutta	5	8	1
(Mod)GenLawson41	(Mod.) Gen. Lawson	4	4	1
(Mod)GenLawson42	(Mod.) Gen. Lawson	4	4	2
(Mod)GenLawson43	(Mod.) Gen. Lawson	4	4	3
(Mod)GenLawson44	(Mod.) Gen. Lawson	5	4	4
(Mod)GenLawson45	(Mod.) Gen. Lawson	6	4	5
PEC423	Predictor-corrector	4	2	3
PECEC433	Predictor-corrector	4	3	3
PEC524	Predictor-corrector	5	2	4
PECEC534	Predictor-corrector	5	3	4
PEC625	Predictor-corrector	6	2	5
PECEC635	Predictor-corrector	6	3	5
PEC726	Predictor-corrector	7	2	6
PECEC736	Predictor-corrector	7	3	6

Comparison of exponential integrators (Q2)

Method	Type	Order	Stages s	Steps q
ABNørsett4	ETD Adams-Bashfort	4	1	4
ABNørsett5	ETD Adams-Bashfort	5	1	5
ABNørsett6	ETD Adams-Bashfort	6	1	6
Friedli (VRK4)	ETD Runge-Kutta	4	4	1
Strehmel-Weiner	ETD Runge-Kutta	4	4	1
Cox-Matthews (ETDRK4)	ETD Runge-Kutta	4	4	1
Krogstad (ETDRK4-B)	ETD Runge-Kutta	4	4	1
Minchev	ETD Runge-Kutta	4	4	1
Hochbruck-Ostermann	ETD Runge-Kutta	4	5	1
Luan-Ostermann (EXPRK5S8)	ETD Runge-Kutta	5	8	1
(Mod)GenLawson41	(Mod.) Gen. Lawson	4	4	1
(Mod)GenLawson42	(Mod.) Gen. Lawson	4	4	2
(Mod)GenLawson43	(Mod.) Gen. Lawson	4	4	3
(Mod)GenLawson44	(Mod.) Gen. Lawson	5	4	4
(Mod)GenLawson45	(Mod.) Gen. Lawson	6	4	5
PEC423	Predictor-corrector	4	2	3
PECEC433	Predictor-corrector	4	3	3
PEC524	Predictor-corrector	5	2	4
PECEC534	Predictor-corrector	5	3	4
PEC625	Predictor-corrector	6	2	5
PECEC635	Predictor-corrector	6	3	5
PEC726	Predictor-corrector	7	2	6
PECEC736	Predictor-corrector	7	3	6

- For certain problems, several methods of orders 5 to 7 are slightly more efficient

Comparison of exponential integrators (Q2)

Method	Type	Order	Stages s	Steps q
ABNørsett4	ETD Adams-Bashfort	4	1	4
ABNørsett5	ETD Adams-Bashfort	5	1	5
ABNørsett6	ETD Adams-Bashfort	6	1	6
Friedli (VRK4)	ETD Runge-Kutta	4	4	1
Strehmel-Weiner	ETD Runge-Kutta	4	4	1
Cox-Matthews (ETDRK4)	ETD Runge-Kutta	4	4	1
Krogstad (ETDRK4-B)	ETD Runge-Kutta	4	4	1
Minchev	ETD Runge-Kutta	4	4	1
Hochbruck-Ostermann	ETD Runge-Kutta	4	5	1
Luan-Ostermann (EXPRK5S8)	ETD Runge-Kutta	5	8	1
(Mod)GenLawson41	(Mod.) Gen. Lawson	4	4	1
(Mod)GenLawson42	(Mod.) Gen. Lawson	4	4	2
(Mod)GenLawson43	(Mod.) Gen. Lawson	4	4	3
(Mod)GenLawson44	(Mod.) Gen. Lawson	5	4	4
(Mod)GenLawson45	(Mod.) Gen. Lawson	6	4	5
PEC423	Predictor-corrector	4	2	3
PECEC433	Predictor-corrector	4	3	3
PEC524	Predictor-corrector	5	2	4
PECEC534	Predictor-corrector	5	3	4
PEC625	Predictor-corrector	6	2	5
PECEC635	Predictor-corrector	6	3	5
PEC726	Predictor-corrector	7	2	6
PECEC736	Predictor-corrector	7	3	6

- For certain problems, several methods of orders 5 to 7 are slightly more efficient
- However, the gains are relatively small and they are sometimes less stable

Comparison of exponential integrators (Q2)

Method	Type	Order	Stages s	Steps q
ABNørsett4	ETD Adams-Bashfort	4	1	4
ABNørsett5	ETD Adams-Bashfort	5	1	5
ABNørsett6	ETD Adams-Bashfort	6	1	6
Friedli (VRK4)	ETD Runge-Kutta	4	4	1
Strehmel-Weiner	ETD Runge-Kutta	4	4	1
Cox-Matthews (ETDRK4)	ETD Runge-Kutta	4	4	1
Krogstad (ETDRK4-B)	ETD Runge-Kutta	4	4	1
Minchev	ETD Runge-Kutta	4	4	1
Hochbruck-Ostermann	ETD Runge-Kutta	4	5	1
Luan-Ostermann (EXPRK5S8)	ETD Runge-Kutta	5	8	1
(Mod)GenLawson41	(Mod.) Gen. Lawson	4	4	1
(Mod)GenLawson42	(Mod.) Gen. Lawson	4	4	2
(Mod)GenLawson43	(Mod.) Gen. Lawson	4	4	3
(Mod)GenLawson44	(Mod.) Gen. Lawson	5	4	4
(Mod)GenLawson45	(Mod.) Gen. Lawson	6	4	5
PEC423	Predictor-corrector	4	2	3
PECEC433	Predictor-corrector	4	3	3
PEC524	Predictor-corrector	5	2	4
PECEC534	Predictor-corrector	5	3	4
PEC625	Predictor-corrector	6	2	5
PECEC635	Predictor-corrector	6	3	5
PEC726	Predictor-corrector	7	2	6
PECEC736	Predictor-corrector	7	3	6

- For certain problems, several methods of orders 5 to 7 are slightly more efficient
- However, the gains are relatively small and they are sometimes less stable
- Hard to do much better than ETDRK4

Adaptive computation (Q3)

Adaptive computation (Q3)

- The algorithm is adaptive in both space and time

Adaptive computation (Q3)

- The algorithm is adaptive in both space and time
- Before the time-stepping loop starts, the coefficients with N points and time-step dt, and with N and $dt/2$, are computed

Adaptive computation (Q3)

- The algorithm is adaptive in both space and time
- Before the time-stepping loop starts, the coefficients with N points and time-step dt, and with N and $dt/2$, are computed
- Algorithm:

Adaptive computation (Q3)

- The algorithm is adaptive in both space and time
- Before the time-stepping loop starts, the coefficients with N points and time-step dt, and with N and $dt/2$, are computed
- Algorithm:

(1) Do one step with N and dt

Adaptive computation (Q3)

- The algorithm is adaptive in both space and time
- Before the time-stepping loop starts, the coefficients with N points and time-step dt, and with N and $dt/2$, are computed
- Algorithm:

(1) Do one step with N and dt

(2) Look at the Fourier coefficients, if they are small enough go on, else increase N (e.g., $N = 2N$ in 1D), recompute the coefficients and go to (1)

Adaptive computation (Q3)

- The algorithm is adaptive in both space and time
- Before the time-stepping loop starts, the coefficients with N points and time-step dt, and with N and $dt/2$, are computed
- Algorithm:

(1) Do one step with N and dt

(2) Look at the Fourier coefficients, if they are small enough go on, else increase N (e.g., $N = 2N$ in 1D), recompute the coefficients and go to (1)

(3) Do two steps with N points and time-step $dt/2$

Adaptive computation (Q3)

- The algorithm is adaptive in both space and time
- Before the time-stepping loop starts, the coefficients with N points and time-step dt, and with N and $dt/2$, are computed
- Algorithm:

(1) Do one step with N and dt

(2) Look at the Fourier coefficients, if they are small enough go on, else increase N (e.g., $N = 2N$ in 1D), recompute the coefficients and go to (1)

(3) Do two steps with N points and time-step $dt/2$

(4) Compare solutions, if the difference is small enough $t = t + dt$ and go to (1), else set $dt = dt/2$, recompute the coefficients and go to (1)

Adaptive computation (Q3)

- The algorithm is adaptive in both space and time
- Before the time-stepping loop starts, the coefficients with N points and time-step dt, and with N and $dt/2$, are computed
- Algorithm:

(1) Do one step with N and dt

(2) Look at the Fourier coefficients, if they are small enough go on, else increase N (e.g., $N = 2N$ in 1D), recompute the coefficients and go to (1)

(3) Do two steps with N points and time-step $dt/2$

(4) Compare solutions, if the difference is small enough $t = t + dt$ and go to (1), else set $dt = dt/2$, recompute the coefficients and go to (1)

- The computation of the coefficients is expensive, so dt is increased every 50 successful time-step

Adaptive computation (Q3)

- The algorithm is adaptive in both space and time
- Before the time-stepping loop starts, the coefficients with N points and time-step dt, and with N and $dt/2$, are computed
- Algorithm:

(1) Do one step with N and dt

(2) Look at the Fourier coefficients, if they are small enough go on, else increase N (e.g., $N = 2N$ in 1D), recompute the coefficients and go to (1)

(3) Do two steps with N points and time-step $dt/2$

(4) Compare solutions, if the difference is small enough $t = t + dt$ and go to (1), else set $dt = dt/2$, recompute the coefficients and go to (1)

- The computation of the coefficients is expensive, so dt is increased every 50 successful time-step
- Default in 1D, optional in 2D and 3D (might be slow on certain laptop)

Examples of 1D stiff PDEs (Q1)

- **Allen-Cahn** (AC)

  Meta-stability, phase separation

$$u_t = 5 \cdot 10^{-2} u_{xx} + u - u^3$$

- **Viscous Burgers** (Burg)

  Shock formation, turbulence

$$u_t = 10^{-3} u_{xx} - \frac{1}{2}(u^2)_x$$

- **Cahn-Hilliard** (CH)

  Meta-stability, phase separation

$$u_t = -10^{-2}(u_{xx} + 10^{-3} u_{xxxx} + (u^3)_{xx})$$

- **Gray-Scott** (GS)

  Pulse splitting, chemical reaction

$$\begin{cases} u_t = u_{xx} + 2 \cdot 10^{-2}(1-u) - uv^2, \\ v_t = 10^{-2} v_{xx} - 8.62 \cdot 10^{-2} v + uv^2 \end{cases}$$

- **Kuramoto-Sivashinsky** (KS)

  Chaos, thermal instabilities

$$u_t = -u_{xx} - u_{xxxx} - \frac{1}{2}(u^2)_x$$

- **Nonlinear Schrödinger** (NLS)

  Breathers, propagation of light

$$u_t = iu_{xx} - iu|u|^2$$

- **Ohta-Kawaski** (OK)

  Pattern formation, copolymers

$$u_t = -u_{xx} - 10^{-2} u_{xxxx} - 4\bar{u} + (u^3)_{xx},$$
$$\bar{u} = u - \int u(t,x)dx$$

UNIVERSITY OF OXFORD

Examples of 2D and 3D stiff PDEs (Q1)

- **GInzburg-Landau** (`GL2`, `GL3`)

  Pattern formation, superconductivity

  $$u_t = \Delta u + u - (1+1.3i)u|u|^2$$

- **Schnakenberg** (`Schnak2`, `Schnak3`)

  Pattern formation, chemical reaction

  $$\begin{cases} u_t = \Delta u + 0.1 - u + u^2 v, \\ v_t = 10\Delta v + 0.9 - u^2 v \end{cases}$$

- **Gray-Scott** (`GS2`, `GS3`)

  Pattern formation, chemical reaction

  $$\begin{cases} u_t = 2 \cdot 10^{-5}\Delta u + F(1-u) - uv^2, \\ v_t = 10^{-5}\Delta v - (F+K)v + uv^2, \end{cases}$$

  with $F = 3.5 \cdot 10^{-2}, \ K = 6 \cdot 10^{-2}$

- **Swift-Hohenberg** (`SH2`, `SH3`)

  Pattern formation, Rayleigh-Bénard convection

  $$u_t = -2\Delta u - \Delta^2 u - 0.9u + u^2 - u^3$$

Conclusion

Conclusion

- Exponential integrators for the high-accuracy solution of stiff PDEs in 1D, 2D and 3D are competitive stiff solvers

Conclusion

- Exponential integrators for the high-accuracy solution of stiff PDEs in 1D, 2D and 3D are competitive stiff solvers
- High-order methods can be more efficient for certain problems, but in general, hard to do much better than ETDRK4

Conclusion

- Exponential integrators for the high-accuracy solution of stiff PDEs in 1D, 2D and 3D are competitive stiff solvers
- High-order methods can be more efficient for certain problems, but in general, hard to do much better than ETDRK4
- Adaptivity in space and time is possible and its cost is negligible in 1D

Conclusion

- Exponential integrators for the high-accuracy solution of stiff PDEs in 1D, 2D and 3D are competitive stiff solvers
- High-order methods can be more efficient for certain problems, but in general, hard to do much better than ETDRK4
- Adaptivity in space and time is possible and its cost is negligible in 1D
- Future work includes PDEs on the sphere (with `spherefun`)