A Fast and Stable Explicit Operator Splitting Method for Phase-Field Models

Zhuolin Qu

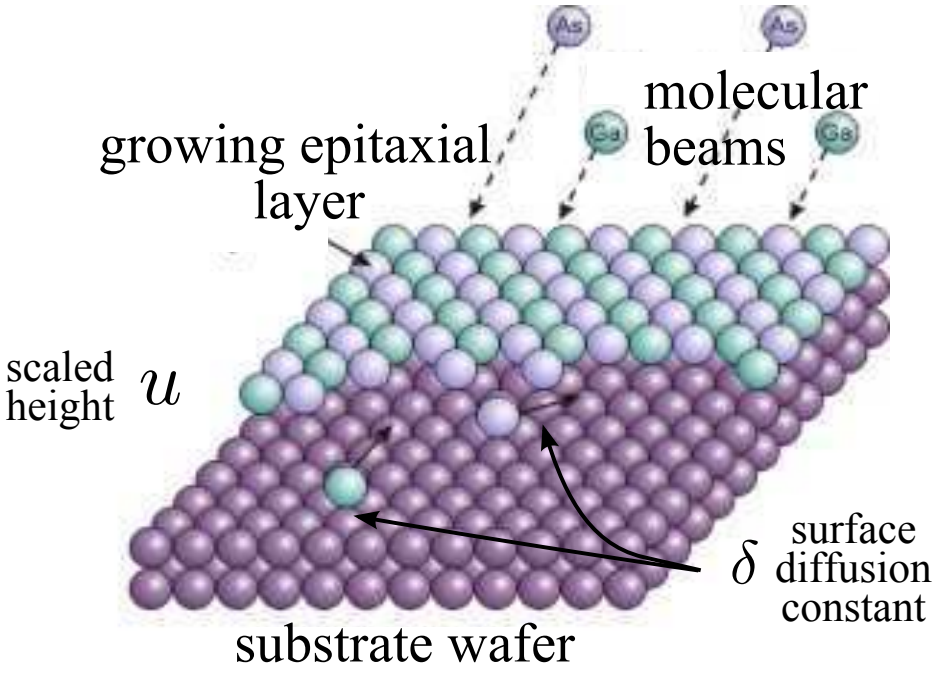
Mathematics Department
Tulane University

Alexander Kurganov, Tulane University, USA
Tao Tang, Hong Kong Baptist University, Hong Kong
Yuanzhen Cheng, Tulane University, USA

Outlines

Phase Field Models: mathematical models for interfacial problems

- Thin film epitaxy: the deposition of a crystalline overlayer on a crystalline substrate

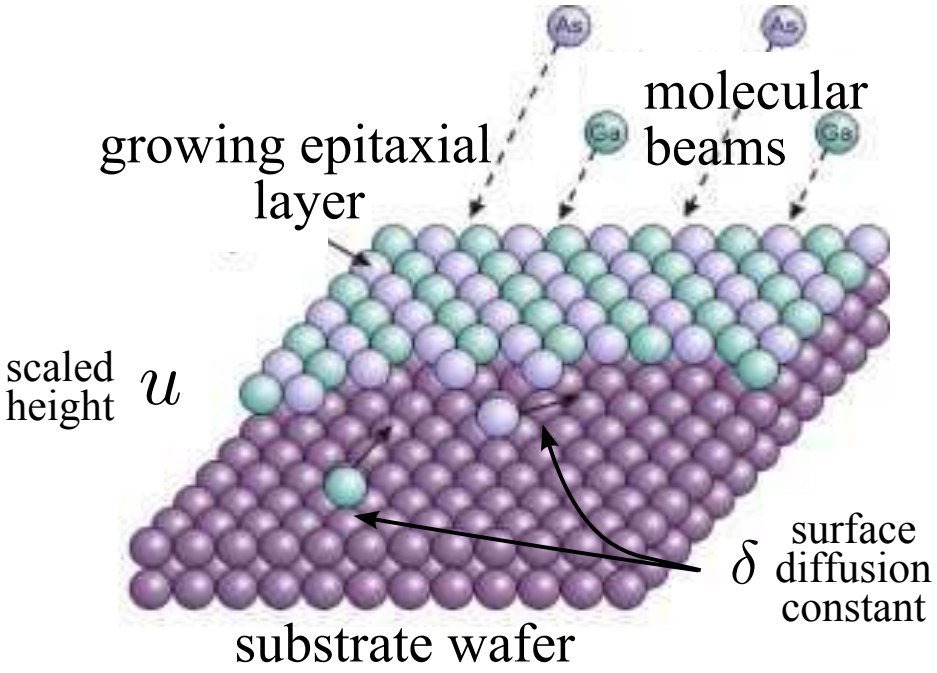


$\Rightarrow$ molecular beam epitaxy (MBE) equation with slop selection

$$u_t = -\delta \Delta^2 u - \nabla \cdot [(1 - |\nabla u|^2)\nabla u], \quad \boldsymbol{x} \in \Omega \subset \mathbb{R}^2$$

Phase Field Models: mathematical models for interfacial problems

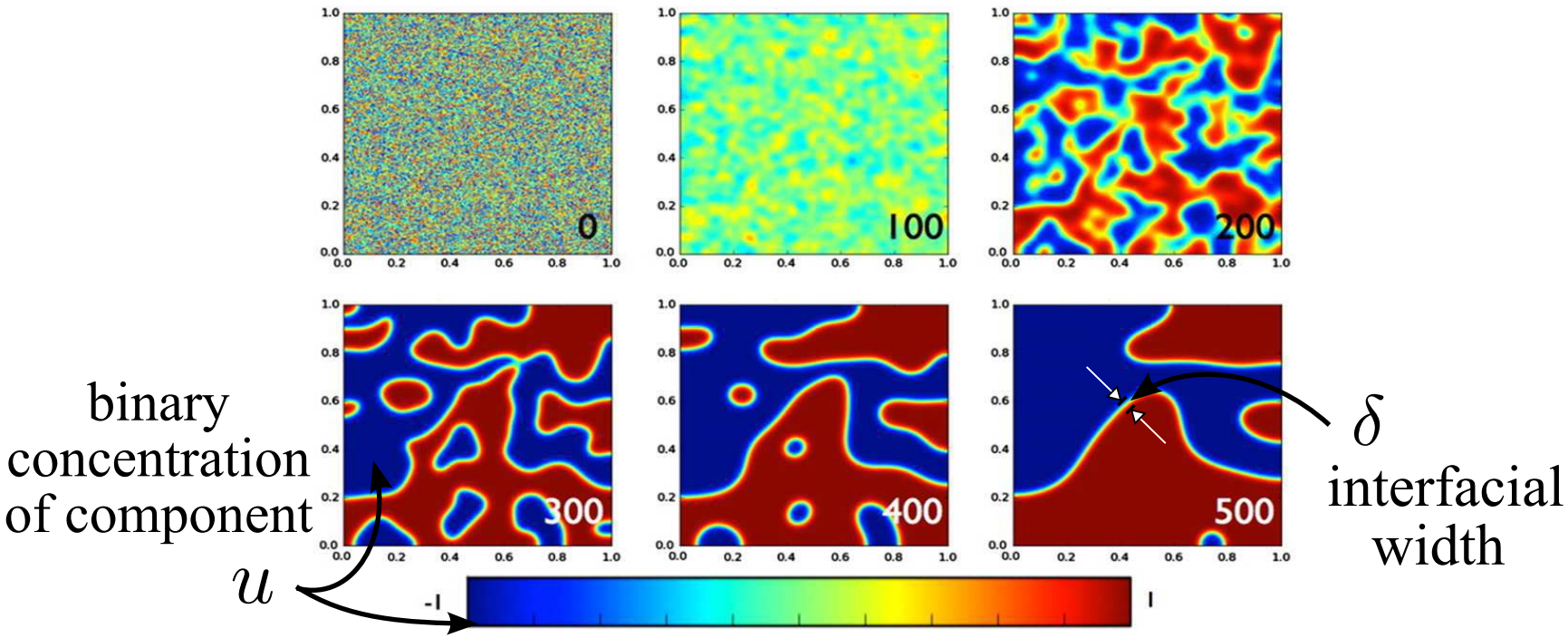
- Thin film epitaxy: the deposition of a crystalline overlayer on a crystalline substrate



$\Rightarrow$ molecular beam epitaxy (MBE) equation with slop selection

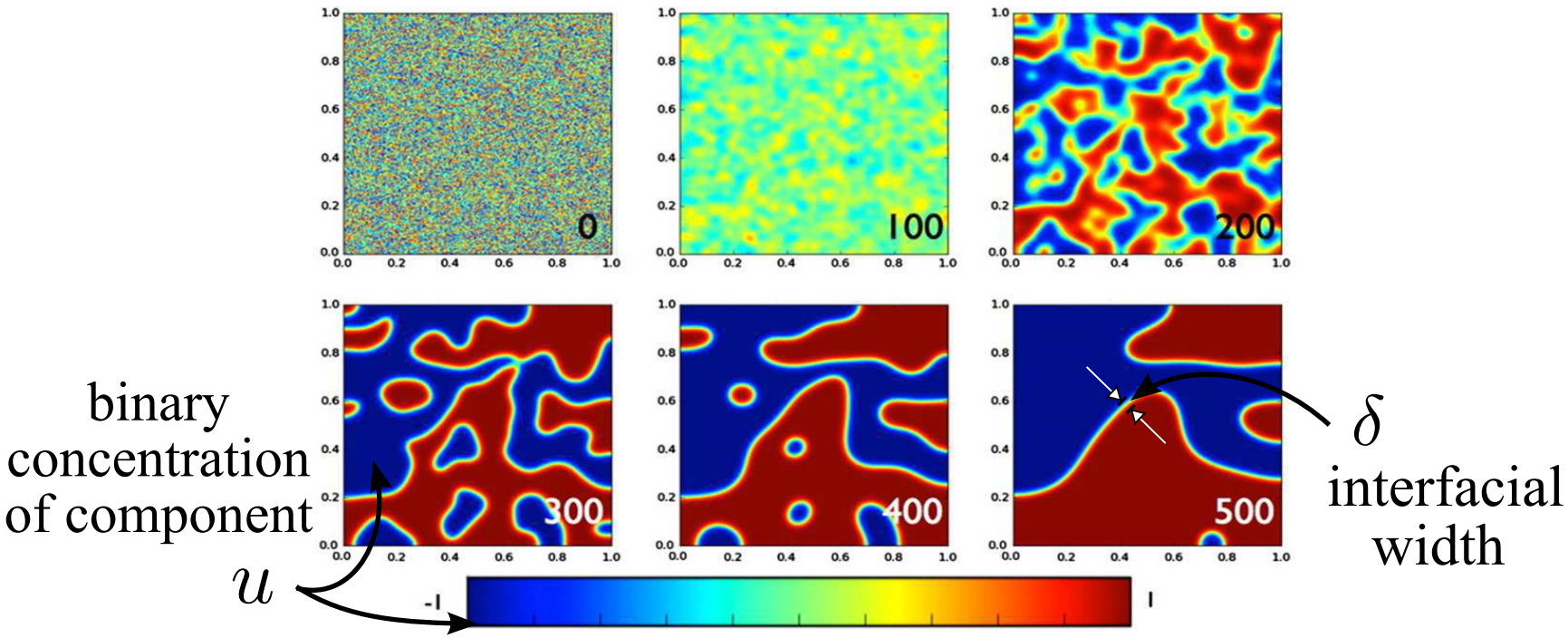
$$u_t = -\delta\Delta^2 u - \nabla\cdot[(1-|\nabla u|^2)\nabla u], \quad \boldsymbol{x}\in\Omega\subset\mathbb{R}^2$$

- Phase separation: two components of a binary fluid spontaneously separate and form domains pure in each component



T. Cool et al, *Gibbs: Phase equilibria and symbolic computation of thermodynamic properties*, Calphad, 34 (2010), pp. 393–404

- Phase separation: two components of a binary fluid spontaneously separate and form domains pure in each component



$\Rightarrow$ Cahn-Hilliard (CH) equation:

$$u_t = -\delta \Delta^2 u + \Delta(u^3 - u), \quad \boldsymbol{x} \in \Omega \subset \mathbb{R}^2$$

T. Cool et al, *Gibbs: Phase equilibria and symbolic computation of thermodynamic properties*, Calphad, 34 (2010), pp. 393–404

Energy Functionals

$$MBE: \quad u_t = -\delta\Delta^2 u - \nabla\cdot[(1-|\nabla u|^2)\nabla u]$$

$$CH: \quad u_t = -\delta\Delta^2 u + \Delta(u^3 - u)$$

Energy Functionals

An important feature of these two equations is that they can be viewed as the gradient flow of energy functionals:

$$MBE: \qquad u_t = -\delta\Delta^2 u - \nabla\cdot[(1-|\nabla u|^2)\nabla u]$$

$$E(u) = \int_\Omega \left[\frac{\delta}{2}|\Delta u|^2 + \frac{1}{4}(|\nabla u|^2 - 1)^2\right] d\boldsymbol{x}$$

$$CH: \qquad u_t = -\delta\Delta^2 u + \Delta(u^3 - u)$$

$$E(u) = \int_\Omega \left[\frac{\delta}{2}|\nabla u|^2 + \frac{1}{4}(u^2 - 1)^2\right] d\boldsymbol{x}$$

Energy Functionals

An important feature of these two equations is that they can be viewed as the gradient flow of energy functionals:

$$MBE: \qquad u_t = -\delta\Delta^2 u - \nabla\cdot[(1-|\nabla u|^2)\nabla u]$$

$$E(u) = \int_\Omega \left[\frac{\delta}{2}|\Delta u|^2 + \frac{1}{4}(|\nabla u|^2 - 1)^2\right] d\boldsymbol{x}$$

$$CH: \qquad u_t = -\delta\Delta^2 u + \Delta(u^3 - u)$$

$$E(u) = \int_\Omega \left[\frac{\delta}{2}|\nabla u|^2 + \frac{1}{4}(u^2 - 1)^2\right] d\boldsymbol{x}$$

As it has been shown in [Cahn, Hillard; 1958] [Li, Liu; 2003], both energy functionals decay in time:

$$E(u(t)) \le E(u(s)),\ \forall t \ge s$$

Numerical Challenges

$$(MBE) \quad u_t = -\delta\Delta^2 u - \nabla\cdot[(1-|\nabla u|^2)\nabla u]$$

$$(CH) \quad u_t = -\delta\Delta^2 u + \Delta(u^3 - u)$$

- Severe Timestep Restriction (accuracy)
  To accurately resolve dynamics, the perturbed biharmonic operator $\delta\Delta^2(\cdot)$, which is involved in the governing equations, may lead to severe restriction on numerical timestep selection.
- Long-Time Simulations (efficiency)
  Numerical simulations of phase-field models require long time computations to attain the steady states (equilibria) of the corresponding phase-field models.
- Nonlinear Energy Stability
  Strong nonlinearities within energy are intrinsic in phase-field models. Violating the energy stability may lead to nonphysical oscillations.

Numerical Challenges

$$(MBE) \quad u_t = -\delta\Delta^2 u - \nabla\cdot[(1-|\nabla u|^2)\nabla u]$$

$$(CH) \quad u_t = -\delta\Delta^2 u + \Delta(u^3 - u)$$

- Severe Timestep Restriction (accuracy)
  To accurately resolve dynamics, the perturbed biharmonic operator $\delta\Delta^2(\cdot)$, which is involved in the governing equations, may lead to severe restriction on numerical timestep selection.
- Long-Time Simulations (efficiency)
  Numerical simulations of phase-field models require long time computations to attain the steady states (equilibria) of the corresponding phase-field models.
- Nonlinear Energy Stability
  Strong nonlinearities within energy are intrinsic in phase-field models. Violating the energy stability may lead to nonphysical oscillations.

Numerical Challenges

$$(MBE) \quad u_t = -\delta\Delta^2 u - \nabla\cdot[(1-|\nabla u|^2)\nabla u]$$

$$(CH) \quad u_t = -\delta\Delta^2 u + \Delta(u^3 - u)$$

- Severe Timestep Restriction (accuracy)
  To accurately resolve dynamics, the perturbed biharmonic operator $\delta\Delta^2(\cdot)$, which is involved in the governing equations, may lead to severe restriction on numerical timestep selection.
- Long-Time Simulations (efficiency)
  Numerical simulations of phase-field models require long time computations to attain the steady states (equilibria) of the corresponding phase-field models.
- Nonlinear Energy Stability
  Strong nonlinearities within energy are intrinsic in phase-field models. Violating the energy stability may lead to nonphysical oscillations.

Numerical Challenges

$$(MBE) \quad u_t = -\delta\Delta^2 u - \nabla\cdot[(1-|\nabla u|^2)\nabla u]$$

$$(CH) \quad u_t = -\delta\Delta^2 u + \Delta(u^3 - u)$$

- Severe Timestep Restriction (accuracy)
  To accurately resolve dynamics, the perturbed biharmonic operator $\delta\Delta^2(\cdot)$, which is involved in the governing equations, may lead to severe restriction on numerical timestep selection.
- Long-Time Simulations (efficiency)
  Numerical simulations of phase-field models require long time computations to attain the steady states (equilibria) of the corresponding phase-field models.
- Nonlinear Energy Stability
  Strong nonlinearities within energy are intrinsic in phase-field models. Violating the energy stability may lead to nonphysical oscillations.

Semi-Implicit Schemes

Explicit schemes usually suffer from severe stability restrictions caused by the presence of high-order derivative terms and do not obey the energy decay property, semi-implicit schemes are widely used.

- [*Xu*, *Tang*; 2006] a combined spectral and large-time steeping method for MBE equation by including an extra stabilization term
- [*He*, *Liu*, *Tang*; 2007] same method was applied to CH equation
- [*Qiao*, *Zhang*, *Tang*; 2011] unconditional energy stable finite-difference schemes with adaptive time-stepping strategy
- [*Zhang*, *Qiao*; 2012] technique was successfully applied in CH equation

Outlines

Explicit Operator Splitting Methods for MBE

MBE equation:

$$u_t = -\delta\Delta^2 u - \nabla\cdot[(1-|\nabla u|^2)\nabla u]$$

linear part

nonlinear part

$$u_t = -\Delta u - \delta\Delta^2 u$$

pseudo-spectral methods

Strang splitting method

$$u_t = \nabla\cdot[|\nabla u|^2\nabla u]$$

semi-discrete finite-difference methods

Strang splitting method

$\mathcal{S}_{\mathcal{L}}$: *exact* solution operator associated with linear part

$\mathcal{S}_{\mathcal{N}}$: *exact* solution operator associated with nonlinear part

Δt: a (small) splitting step

$$u(\mathbf{x}, t+\Delta t) = \mathcal{S}_{\mathcal{L}}(\Delta t/2)\mathcal{S}_{\mathcal{N}}(\Delta t)\mathcal{S}_{\mathcal{L}}(\Delta t/2)u(\mathbf{x}, t)$$

Explicit Operator Splitting Methods for MBE

MBE equation:

$$u_t = -\delta\Delta^2 u - \nabla\cdot[(1-|\nabla u|^2)\nabla u]$$

linear part

$$u_t = -\Delta u - \delta\Delta^2 u$$

pseudo-spectral methods

Strang splitting method

nonlinear part

$$u_t = \nabla\cdot[|\nabla u|^2\nabla u]$$

semi-discrete finite-difference methods

Strang splitting method

$\mathcal{S}_{\mathcal{L}}$: *exact* solution operator associated with linear part

$\mathcal{S}_{\mathcal{N}}$: *exact* solution operator associated with nonlinear part

Δt: a (small) splitting step

$$u(\boldsymbol{x}, t+\Delta t) = \mathcal{S}_{\mathcal{L}}(\Delta t/2)\mathcal{S}_{\mathcal{N}}(\Delta t)\mathcal{S}_{\mathcal{L}}(\Delta t/2)u(\boldsymbol{x}, t)$$

Linear Part

$$u_t = -\Delta u - \delta \Delta^2 u$$

Pseudo-Spectral method with fast-fourier transform (FFT):

$$\boxed{\begin{aligned} u_t &= -\Delta u - \delta \Delta^2 u \\ \Omega &= [0, L_x] \times [0, L_y] \end{aligned}} \xrightarrow{\text{FFT}} \boxed{\begin{aligned} \frac{d}{dt}\widehat{u}_{m,\ell}(t) &= (r - \delta r^2)\widehat{u}_{m,\ell}(t) \\ r &= \left(\frac{2\pi m}{L_x}\right)^2 + \left(\frac{2\pi \ell}{L_y}\right)^2 \end{aligned}}$$

$$\xrightarrow{\text{exact solution}}$$

$$\boxed{\widehat{u}_{m,\ell}(t + \Delta t) = e^{(r - \delta r^2)\Delta t}\widehat{u}_{m,\ell}(t)} \xrightarrow{\text{IFFT}} \boxed{\{u_{j,k}(t + \Delta t)\}}$$

Nonlinear Part $u_t = \nabla \cdot [|\nabla u|^2 \nabla u]$

The 1-D version is

$$u_t = (u_x^3)_x$$

We consider a uniform grid and introduce the following $2m^{th}$-order centered-difference approximation of the $\frac{\partial}{\partial x}$ operator:

$$(\psi_x)_j := \sum_{p=-m}^{m} \alpha_p \psi_{j+p} = \psi_x(x_j) + \mathcal{O}((\Delta x)^{2m})$$

Note that:

$$\alpha_0 = 0 \quad \text{and} \quad \alpha_p + \alpha_{-p} = 0, \ p \neq 0$$

For example, when $m = 2$, we obtain a fourth-order centered-difference approximation by taking

$$\alpha_1 = -\alpha_{-1} = \frac{2}{3\Delta x} \quad \alpha_2 = -\alpha_{-2} = -\frac{1}{12\Delta x}$$

Nonlinear Part $u_t = \nabla \cdot [|\nabla u|^2 \nabla u]$

The 1-D version is

$$u_t = (u_x^3)_x$$

We consider a uniform grid and introduce the following $2m^{th}$-order centered-difference approximation of the $\frac{\partial}{\partial x}$ operator:

$$(\psi_x)_j := \sum_{p=-m}^{m} \alpha_p \psi_{j+p} = \psi_x(x_j) + \mathcal{O}((\Delta x)^{2m})$$

Note that:

$$\alpha_0 = 0 \quad \text{and} \quad \alpha_p + \alpha_{-p} = 0, \ p \neq 0$$

For example, when $m = 2$, we obtain a fourth-order centered-difference approximation by taking

$$\alpha_1 = -\alpha_{-1} = \frac{2}{3\Delta x} \quad \alpha_2 = -\alpha_{-2} = -\frac{1}{12\Delta x}$$

Nonlinear Part $u_t = \nabla \cdot [|\nabla u|^2 \nabla u]$

The 1-D version is

$$u_t = (u_x^3)_x$$

We consider a uniform grid and introduce the following $2m^{th}$-order centered-difference approximation of the $\frac{\partial}{\partial x}$ operator:

$$(\psi_x)_j := \sum_{p=-m}^{m} \alpha_p \psi_{j+p} = \psi_x(x_j) + \mathcal{O}((\Delta x)^{2m})$$

Note that:

$$\alpha_0 = 0 \quad \text{and} \quad \alpha_p + \alpha_{-p} = 0, \ p \neq 0$$

For example, when $m = 2$, we obtain a fourth-order centered-difference approximation by taking

$$\alpha_1 = -\alpha_{-1} = \frac{2}{3\Delta x} \quad \alpha_2 = -\alpha_{-2} = -\frac{1}{12\Delta x}$$

$$u_t = (u_x^3)_x$$

We discretize equation using the method of lines as follows:

$$\frac{du_j}{dt}(t) = \sum_{p=-m}^{m} \alpha_p H_{j+p}(t)$$

where $u_j(t)$ denotes the computed point value of the solution at (x_j, t), and

$$H_j(t) := (u_x)_j^3(t)$$

with

$$(u_x)_j(t) := \sum_{p=-m}^{m} \alpha_p u_{j+p}(t)$$

$$u_t = (u_x^3)_x$$

We discretize equation using the method of lines as follows:

$$\frac{du_j}{dt}(t) = \sum_{p=-m}^{m} \alpha_p H_{j+p}(t)$$

where $u_j(t)$ denotes the computed point value of the solution at (x_j, t), and

$$H_j(t) := (u_x)_j^3(t)$$

with

$$(u_x)_j(t) := \sum_{p=-m}^{m} \alpha_p u_{j+p}(t)$$

$$u_t = (u_x^3)_x$$

We discretize equation using the method of lines as follows:

$$\frac{du_j}{dt}(t) = \sum_{p=-m}^{m} \alpha_p H_{j+p}(t)$$

where $u_j(t)$ denotes the computed point value of the solution at (x_j, t), and

$$H_j(t) := (u_x)_j^3(t)$$

with

$$(u_x)_j(t) := \sum_{p=-m}^{m} \alpha_p u_{j+p}(t)$$

For 2-D case:

$$u_t = \nabla \cdot [|\nabla u|^2 \nabla u]$$

We consider a uniform grid, and introduce the following $2m^{th}$-order centered-difference approximation of the $\frac{\partial}{\partial x}$ and $\frac{\partial}{\partial y}$ operators:

$$(\psi_x)_{j,k} := \sum_{p=-m}^{m} \alpha_p \psi_{j+p,k} = \psi_x(x_j, y_k) + \mathcal{O}((\Delta x)^{2m})$$

$$(\psi_y)_{j,k} := \sum_{p=-m}^{m} \beta_p \psi_{j,k+p} = \psi_y(x_j, y_k) + \mathcal{O}((\Delta y)^{2m})$$

Note that:

$$\alpha_0 = 0 \quad \text{and} \quad \alpha_p + \alpha_{-p} = 0,\ \beta_p + \beta_{-p} = 0,\ p \neq 0$$

$2m^{th}$-order semi-discrete finite-difference schemes read:

$$\frac{du_{j,k}}{dt} = \sum_{p=-m}^{m} \alpha_p H^x_{j+p,k} + \sum_{p=-m}^{m} \beta_p H^y_{j,k+p}$$

where

$$H^x_{j,k} := (u_x)^3_{j,k} + (u_y)^2_{j,k}(u_x)_{j,k} \quad \text{and} \quad H^y_{j,k} := (u_y)^3_{j,k} + (u_x)^2_{j,k}(u_y)_{j,k}$$

with

$$(u_x)_{j,k} := \sum_{p=-m}^{m} \alpha_p u_{j+p,k} \quad \text{and} \quad (u_y)_{j,k} := \sum_{p=-m}^{m} \beta_p u_{j,k+p}$$

Physical Property?

- Mass Conservation: automatically satisfied by using the flux form
- Energy Decay:

$$E(u) = \int_\Omega \left[\frac{\delta}{2}|\Delta u|^2 + \frac{1}{4}(|\nabla u|^2 - 1)^2 \right] d\boldsymbol{x} = E_{\mathcal{N}}(u) + E_{\mathcal{L}}(u)$$

where

$$E_{\mathcal{L}}(u) = \int_\Omega \left(\frac{\delta}{2}|\Delta u|^2 - \frac{1}{2}|\nabla u|^2 + \frac{1}{4} \right) d\boldsymbol{x}$$

$$E_{\mathcal{N}}(u) = \frac{1}{4} \int_\Omega |\nabla u|^4 \, d\boldsymbol{x}$$

Physical Property?

- Mass Conservation: automatically satisfied by using the flux form
- Energy Decay:

$$E(u) = \int_\Omega \left[\frac{\delta}{2}|\Delta u|^2 + \frac{1}{4}(|\nabla u|^2 - 1)^2 \right] d\boldsymbol{x} = E_{\mathcal{N}}(u) + E_{\mathcal{L}}(u)$$

where

$$E_{\mathcal{L}}(u) = \int_\Omega \left(\frac{\delta}{2}|\Delta u|^2 - \frac{1}{2}|\nabla u|^2 + \frac{1}{4} \right) d\boldsymbol{x}$$

$$E_{\mathcal{N}}(u) = \frac{1}{4} \int_\Omega |\nabla u|^4 \, d\boldsymbol{x}$$

Physical Property?

- Mass Conservation: automatically satisfied by using the flux form
- Energy Decay:

$$E(u) = \int_\Omega \left[\frac{\delta}{2}|\Delta u|^2 + \frac{1}{4}(|\nabla u|^2 - 1)^2\right] d\boldsymbol{x} = E_{\mathcal{N}}(u) + E_{\mathcal{L}}(u)$$

where

$$E_{\mathcal{L}}(u) = \int_\Omega \left(\frac{\delta}{2}|\Delta u|^2 - \frac{1}{2}|\nabla u|^2 + \frac{1}{4}\right) d\boldsymbol{x}$$

$$E_{\mathcal{N}}(u) = \frac{1}{4}\int_\Omega |\nabla u|^4 \, d\boldsymbol{x}$$

Physical Property?

- Mass Conservation: automatically satisfied by using the flux form
- Energy Decay:

$$E(u) = \int_\Omega \left[\frac{\delta}{2} |\Delta u|^2 + \frac{1}{4} (|\nabla u|^2 - 1)^2 \right] d\boldsymbol{x} = E_{\mathcal{N}}(u) + E_{\mathcal{L}}(u)$$

where

$$E_{\mathcal{L}}(u) = \int_\Omega \left(\frac{\delta}{2} |\Delta u|^2 - \frac{1}{2} |\nabla u|^2 + \frac{1}{4} \right) d\boldsymbol{x}$$

$$E_{\mathcal{N}}(u) = \frac{1}{4} \int_\Omega |\nabla u|^4 \, d\boldsymbol{x}$$

Theorem (Energy Decay Property in 1-D) The semi-discrete schemes satisfy the following energy decay property $\frac{d}{dt}E_{\mathcal{N}}^{\Delta} \leq 0$, where $E_{\mathcal{N}}^{\Delta}$ is a 1-D discrete version of the energy functional

$$E_{\mathcal{N}}^{\Delta} := \frac{1}{4}\sum_{j}(u_x)_j^4 \Delta x.$$

Theorem (Energy Decay Property in 2-D) The semi-discrete schemes satisfy the following energy decay property $\frac{d}{dt}E_{\mathcal{N}}^{\Delta} \leq 0$, where $E_{\mathcal{N}}^{\Delta}$ is a 2-D discrete version of the energy functional:

$$E_{\mathcal{N}}^{\Delta} := \frac{1}{4}\sum_{j}|\nabla_h u_{j,k}|^4 \Delta x \Delta y$$

with $\nabla_h u_{j,k} := ((u_x)_{j,k}, (u_y)_{j,k})^T$.

Explicit Operator Splitting Methods for CH

CH equation:

$$u_t = -\delta\Delta^2 u + \Delta(u^3 - u)$$

linear part

nonlinear part

$$u_t = -\Delta u - \delta\Delta^2 u$$

pseudo-spectral methods

Strang splitting method

$$u_t = \Delta(u^3)$$

semi-discrete finite-difference methods

Strang splitting method

$\mathcal{S}_{\mathcal{L}}$: *exact* solution operator associated with linear part

$\mathcal{S}_{\mathcal{N}}$: *exact* solution operator associated with nonlinear part

Δt: a (small) splitting step

$$u(\boldsymbol{x}, t + \Delta t) = \mathcal{S}_{\mathcal{L}}(\Delta t/2)\mathcal{S}_{\mathcal{N}}(\Delta t)\mathcal{S}_{\mathcal{L}}(\Delta t/2)u(\boldsymbol{x}, t)$$

Nonlinear Part $u_t = \Delta(u^3)$

We use the same grids and the same $2m^{th}$-order discrete approximation of the $\frac{\partial}{\partial x}$ and $\frac{\partial}{\partial y}$ operators.

Then, $2m^{th}$-order semi-discrete finite-difference schemes read:

$$\frac{du_{j,k}}{dt} = \sum_{p=-m}^{m} \alpha_p H^x_{j+p,k} + \sum_{p=-m}^{m} \beta_p H^y_{j,k+p}$$

where

$$H^x_{j,k} := \sum_{p=-m}^{m} \alpha_p u^3_{j+p,k} \quad \text{and} \quad H^y_{j,k} := \sum_{p=-m}^{m} \beta_p u^3_{j,k+p}$$

Energy Decay?

$$E(u) = \int_\Omega \left[\frac{\delta}{2}|\nabla u|^2 + \frac{1}{4}(u^2-1)^2\right] d\boldsymbol{x} = E_{\mathcal{L}}(u) + E_{\mathcal{N}}(u)$$

where

$$E_{\mathcal{L}}(u) = \int_\Omega \left(\frac{\delta}{2}|\nabla u|^2 - \frac{1}{2}u^2 + \frac{1}{4}\right) d\boldsymbol{x} \qquad E_{\mathcal{N}}(u) = \frac{1}{4}\int_\Omega u^4\, d\boldsymbol{x}$$

Theorem (Energy Decay Property) The semi-discrete schemes satisfy the following energy decay property:

$$\frac{d}{dt}E_{\mathcal{N}}^{\Delta} \le 0$$

where $E_{\mathcal{N}}^{\Delta}$ is a 2-D discrete version of the energy functional:

$$E_{\mathcal{N}}^{\Delta} := \frac{1}{4}\sum_j u_{j,k}^4 \Delta x \Delta y$$

ODE Solver

All the obtain ODE systems have to be solved numerically.

- Explicit ODE solvers: typically require timesteps to be $\Delta t_{\mathrm{ODE}} \sim (\Delta x)^2$
- Implicit ODE solvers: can be made unconditionally stable, the accuracy requirements would limit timestep size;
  Moreover, a large nonlinear algebraic system of equations has to be solved at each timestep, implicit methods may not be efficient

Our approach:

explicit third-order large stability domain Runge-Kutta method

ODE Solver

All the obtain ODE systems have to be solved numerically.

- Explicit ODE solvers: typically require timesteps to be $\Delta t_{\mathrm{ODE}} \sim (\Delta x)^2$
- Implicit ODE solvers: can be made unconditionally stable, the accuracy requirements would limit timestep size;
  Moreover, a large nonlinear algebraic system of equations has to be solved at each timestep, implicit methods may not be efficient

Our approach:

explicit third-order large stability domain Runge-Kutta method

ODE Solver

All the obtain ODE systems have to be solved numerically.

- Explicit ODE solvers: typically require timesteps to be $\Delta t_{\mathrm{ODE}} \sim (\Delta x)^2$
- Implicit ODE solvers: can be made unconditionally stable, the accuracy requirements would limit timestep size;
  Moreover, a large nonlinear algebraic system of equations has to be solved at each timestep, implicit methods may not be efficient

Our approach:

explicit third-order large stability domain Runge-Kutta method

DUMKA3 [Medovikov; 1998]

- *Large Stability Domain*:
  It belongs to a class of Runge-Kutta-Chebyshev method and allows one to use much larger timesteps compared with the standard Runge-Kutta methods.
- The explicit form retains simplicity, and embedded formulas permit an efficient stepsize control
- Efficiency can be further improved when the user provides an upper bound on the timestep stability restriction for the forward Euler method Δt_{FE}

$\Rightarrow$ Therefore, we establish such bounds in the following three theorems.

DUMKA3 [Medovikov; 1998]

- *Large Stability Domain*:
  It belongs to a class of Runge-Kutta-Chebyshev method and allows one to use much larger timesteps compared with the standard Runge-Kutta methods.
- The explicit form retains simplicity, and embedded formulas permit an efficient stepsize control
- Efficiency can be further improved when the user provides an upper bound on the timestep stability restriction for the forward Euler method Δt_{FE}

$\Rightarrow$ Therefore, we establish such bounds in the following three theorems.

DUMKA3 [Medovikov; 1998]

- *Large Stability Domain*:
  It belongs to a class of Runge-Kutta-Chebyshev method and allows one to use much larger timesteps compared with the standard Runge-Kutta methods.
- The explicit form retains simplicity, and embedded formulas permit an efficient stepsize control
- Efficiency can be further improved when the user provides an upper bound on the timestep stability restriction for the forward Euler method Δt_{FE}

  $\Rightarrow$ Therefore, we establish such bounds in the following three theorems.

DUMKA3 [Medovikov; 1998]

- *Large Stability Domain*:
  It belongs to a class of Runge-Kutta-Chebyshev method and allows one to use much larger timesteps compared with the standard Runge-Kutta methods.
- The explicit form retains simplicity, and embedded formulas permit an efficient stepsize control
- Efficiency can be further improved when the user provides an upper bound on the timestep stability restriction for the forward Euler method Δt_{FE}

$\Rightarrow$ Therefore, we establish such bounds in the following three theorems.

DUMKA3 [Medovikov; 1998]

- *Large Stability Domain*:
  It belongs to a class of Runge-Kutta-Chebyshev method and allows one to use much larger timesteps compared with the standard Runge-Kutta methods.
- The explicit form retains simplicity, and embedded formulas permit an efficient stepsize control
- Efficiency can be further improved when the user provides an upper bound on the timestep stability restriction for the forward Euler method Δt_{FE}

  $\Rightarrow$ Therefore, we establish such bounds in the following three theorems.

Theorem (Δt_{FE} Bound for 1-D MBE) Assume that the system of ODEs is numerically integrated by the forward Euler method from time t to $t + \Delta t_{\mathrm{FE}}$ and that the following CFL condition holds:

$$\Delta t_{\mathrm{FE}} \leq \frac{1}{am} \cdot \frac{1}{\max_j (u_x)_j^2}$$

$$a := \sum_{p=-m}^{m} \alpha_p^2$$

Then

$$\|u(t + \Delta t_{\mathrm{FE}})\|_{L^2} \leq \|u(t)\|_{L^2}$$

where $\|u(t)\|_{L^2} := \sqrt{\sum_j u_j^2(t)\Delta x}$.

Theorem (Δt_{FE} Bound for 2-D MBE) Assume that the system of ODEs is numerically integrated by the forward Euler method from time t to $t + \Delta t_{\mathrm{FE}}$ and that the following CFL condition holds:

$$\Delta t_{\mathrm{FE}} \leq \frac{1}{4m \cdot \max(a, b)} \cdot \frac{1}{\max\limits_{j,k}\{(u_x)_{j,k}^2, (u_y)_{j,k}^2\}}$$

$$a := \sum_{p=-m}^{m} \alpha_p^2 \qquad b := \sum_{p=-m}^{m} \beta_p^2$$

Then

$$\|u(t + \Delta t_{\mathrm{FE}})\|_{L^2} \leq \|u(t)\|_{L^2}$$

where $\|u(t)\|_{L^2} := \sqrt{\sum_{j,k} u_{j,k}^2(t)\Delta x \Delta y}$.

Theorem (Δt_{FE} Bound for 2-D CH) Assume that the system of ODEs is numerically integrated by the forward Euler method from time t to $t + \Delta t_{\mathrm{FE}}$ and that the following CFL condition holds:

$$\Delta t_{\mathrm{FE}} \leq \frac{1}{6m \cdot \max(a, b)} \cdot \frac{1}{\max\limits_{j,k} u_{j,k}^2}$$

$$a := \sum_{p=-m}^{m} \alpha_p^2 \qquad b := \sum_{p=-m}^{m} \beta_p^2$$

Then

$$\|u(t + \Delta t_{\mathrm{FE}})\|_{L^2} \leq \|u(t)\|_{L^2}$$

where $\|u(t)\|_{L^2} := \sqrt{\sum_{j,k} u_{j,k}^2(t) \Delta x \Delta y}$.

Idea for Choosing Timesteps

The efficiency of splitting methods hinges on its ability to use (relatively) large timesteps

- small Δt: the phase transition occurs and the solution changes quite rapidly
- large Δt: at other times and especially the solution is close to its steady state

Idea for Choosing Timesteps

The efficiency of splitting methods hinges on its ability to use (relatively) large timesteps

- small Δt: the phase transition occurs and the solution changes quite rapidly
- large Δt: at other times and especially the solution is close to its steady state

Idea for Choosing Timesteps

The efficiency of splitting methods hinges on its ability to use (relatively) large timesteps

- small Δt: the phase transition occurs and the solution changes quite rapidly
- large Δt: at other times and especially the solution is close to its steady state

How to measure the solution variation?

Idea for Choosing Timesteps

The efficiency of splitting methods hinges on its ability to use (relatively) large timesteps

- small Δt: the phase transition occurs and the solution changes quite rapidly
- large Δt: at other times and especially the solution is close to its steady state

We define the roughness of solution at time t:

$$w(t) = \sqrt{\frac{1}{|\Omega|}\int_\Omega [u(\boldsymbol{x},t) - \bar{u}(t)]^2\, d\boldsymbol{x}}$$

where

$$\bar{u}(t) = \frac{1}{|\Omega|}\int_\Omega u(\boldsymbol{x},t)\, d\boldsymbol{x}$$

is the mean height at time t.

We adjust the size of splitting steps using the following roughness-dependent monitor function [Qiao, Zhang, Tang; 2011]

$$\Delta t = \max\left(\Delta t_{\min},\ \frac{\Delta t_{\max}}{\sqrt{1+\alpha|w'(t)|^2}}\right), \quad \alpha = \text{constant}$$

- $\Delta t_{\min}$: the smallest possible splitting step
  $\Delta t_{\min} = \delta/100$ for the MBE equations
  $\Delta t_{\min} = \delta/10$ for the CH equation
- $\Delta t_{\max}$: the largest allowed splitting step
- α: positively adaption constant

Large $|w'(t)| \Rightarrow$ small Δt: quick motion of the structural transition
Small $|w'(t)| \Rightarrow$ large Δt: slow film growth or slow phase interface motion

a significant reduction of CPU time ($3 \sim 6$ more efficient)
without affect the accuracy of the computed solutions

We adjust the size of splitting steps using the following roughness-dependent monitor function [Qiao, Zhang, Tang; 2011]

$$\Delta t = \max\left(\Delta t_{\min},\ \frac{\Delta t_{\max}}{\sqrt{1+\alpha|w'(t)|^2}}\right), \quad \alpha = \text{constant}$$

- $\Delta t_{\min}$: the smallest possible splitting step
  $\Delta t_{\min} = \delta/100$ for the MBE equations
  $\Delta t_{\min} = \delta/10$ for the CH equation
- $\Delta t_{\max}$: the largest allowed splitting step
- α: positively adaption constant

Large $|w'(t)| \Rightarrow$ small Δt: quick motion of the structural transition
Small $|w'(t)| \Rightarrow$ large Δt: slow film growth or slow phase interface motion

a significant reduction of CPU time ($3 \sim 6$ more efficient)
without affect the accuracy of the computed solutions

We adjust the size of splitting steps using the following roughness-dependent monitor function [Qiao, Zhang, Tang; 2011]

$$\Delta t = \max\left(\Delta t_{\min},\ \frac{\Delta t_{\max}}{\sqrt{1+\alpha|w'(t)|^2}}\right), \quad \alpha = \text{constant}$$

- $\Delta t_{\min}$: the smallest possible splitting step
  $\Delta t_{\min} = \delta/100$ for the MBE equations
  $\Delta t_{\min} = \delta/10$ for the CH equation
- $\Delta t_{\max}$: the largest allowed splitting step
- α: positively adaption constant

Large $|w'(t)| \Rightarrow$ small Δt: quick motion of the structural transition
Small $|w'(t)| \Rightarrow$ large Δt: slow film growth or slow phase interface motion

a significant reduction of CPU time ($3 \sim 6$ more efficient)
without affect the accuracy of the computed solutions

Outlines

Numerical Examples

We illustrate the performance of our fast and stable explicit operator splitting method on several 1-D and 2-D examples. We use

- fourth-order finite-difference schemes
- both constant and adaptive splitting steps are employed to obtain numerical solutions

Example 1 — One-Dimensional Morphological Instability

We first consider the 1-D MBE equation

$$u_t = (u_x^3)_x - u_{xx} - u_{xxxx}$$

subject to the initial condition

$$u(x,0) = 0.1(\sin\frac{\pi x}{2} + \sin\frac{2\pi x}{3} + \sin \pi x), \quad x \in [0, 12]$$

This example was studied in [Li, Liu; 2003] to observe the morphological instability due to the nonlinear interaction.

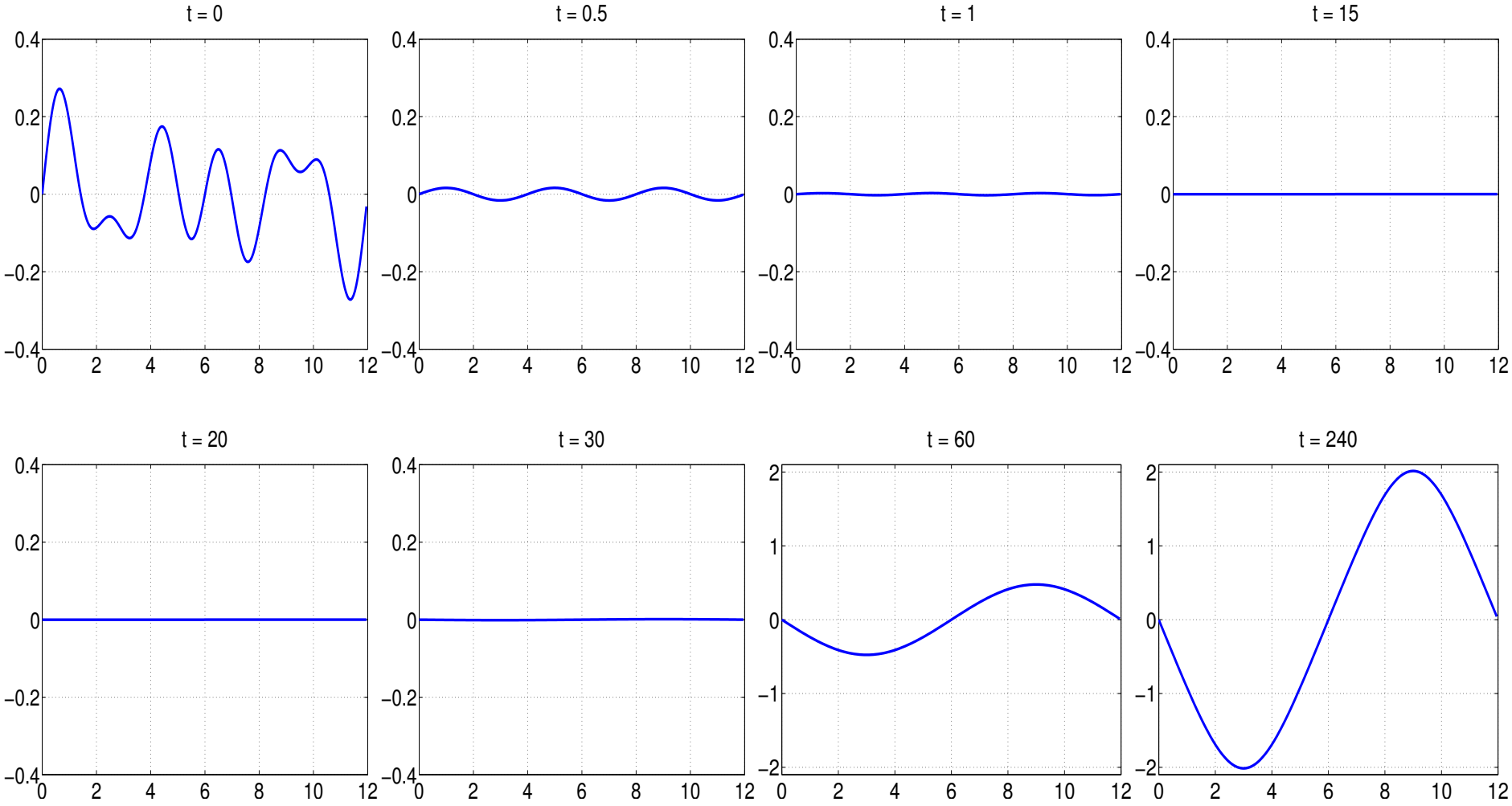


Figure: Example 1: u computed with $\Delta t = 10^{-1}$.

Compared to the results reported in [Li, Liu; 2003], our steady state is in a good agreement with the one obtained there, while the "buffering" time evolution is very different.

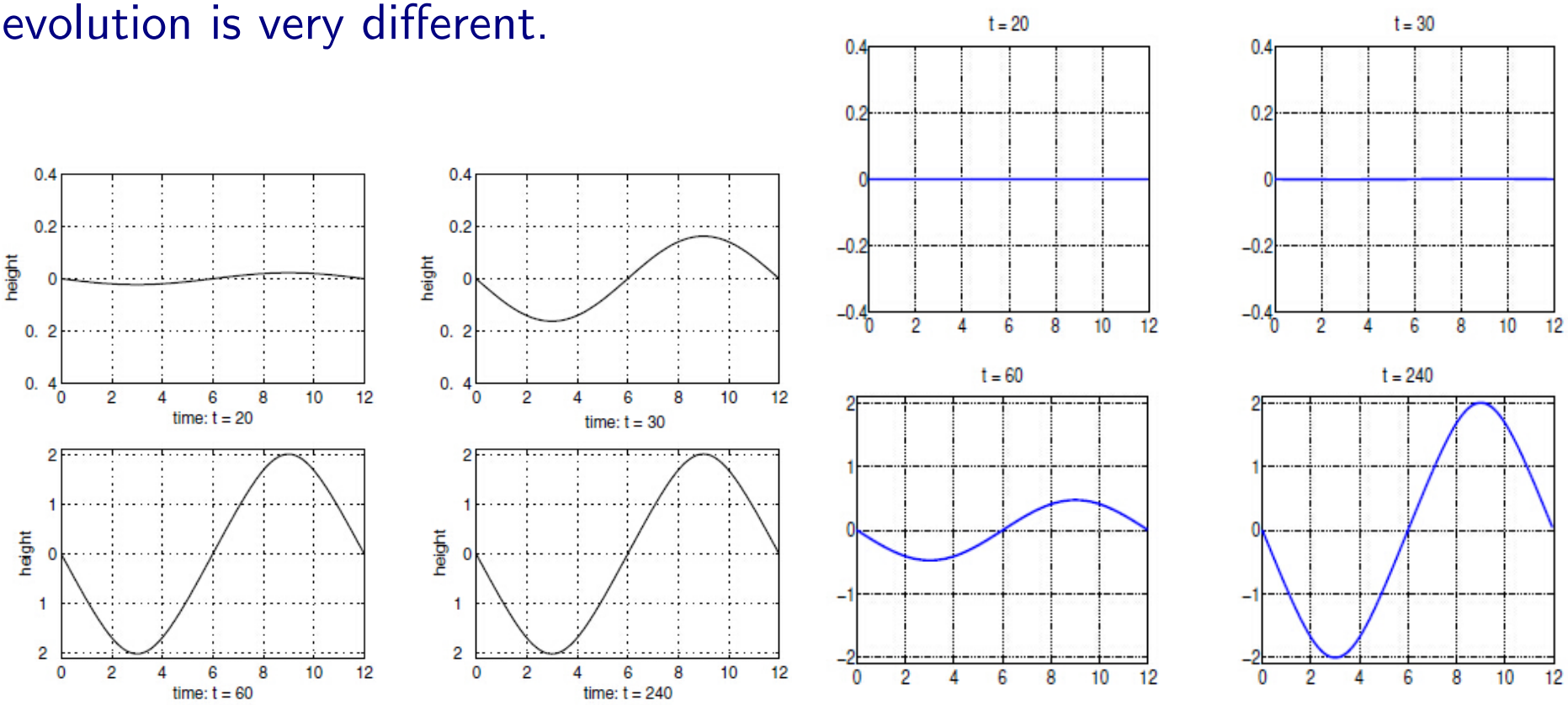


Figure: Left: [Li, Liu; 2003]; Right: our computation.

Difference in "buffering" time?

Compared to the results reported in [Li, Liu; 2003], our steady state is in a good agreement with the one obtained there, while the "buffering" time evolution is very different.

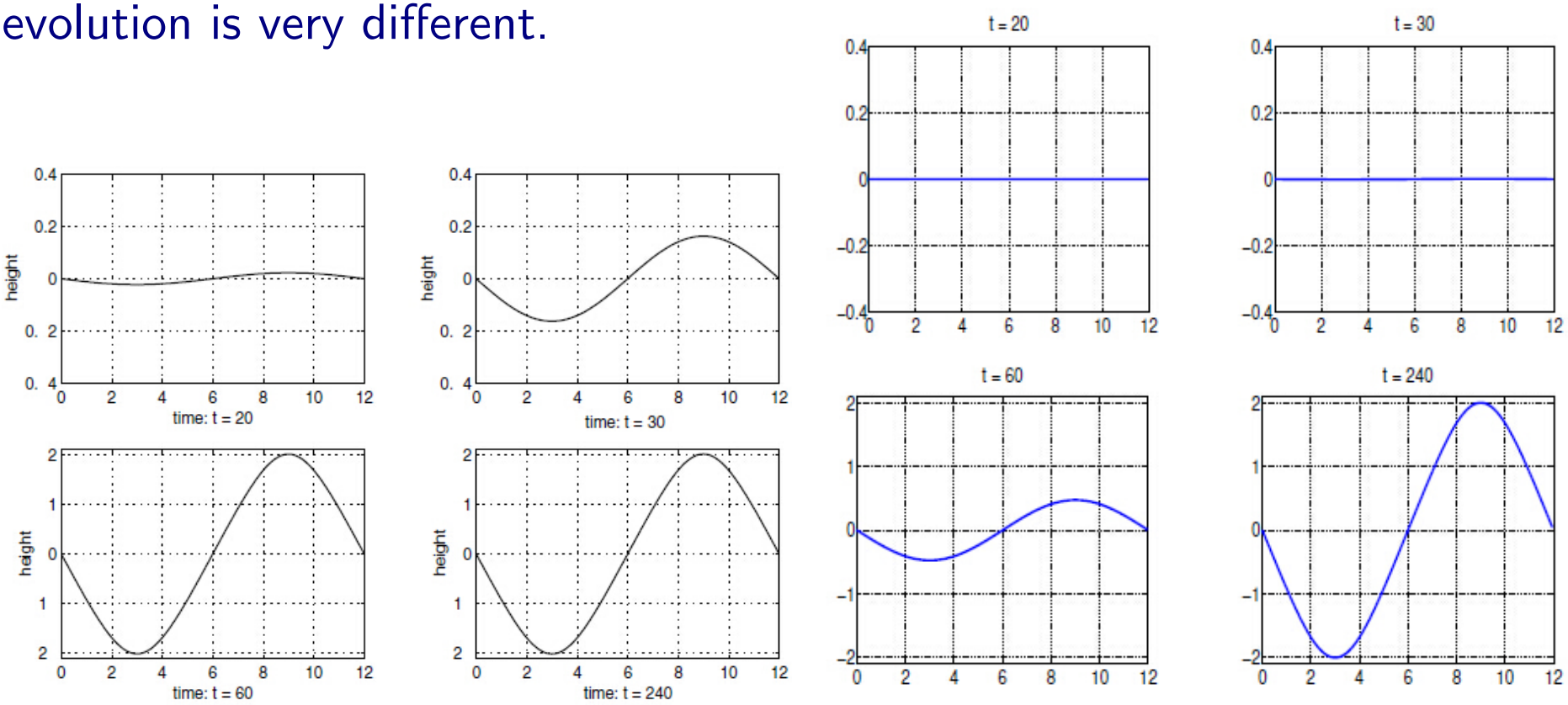
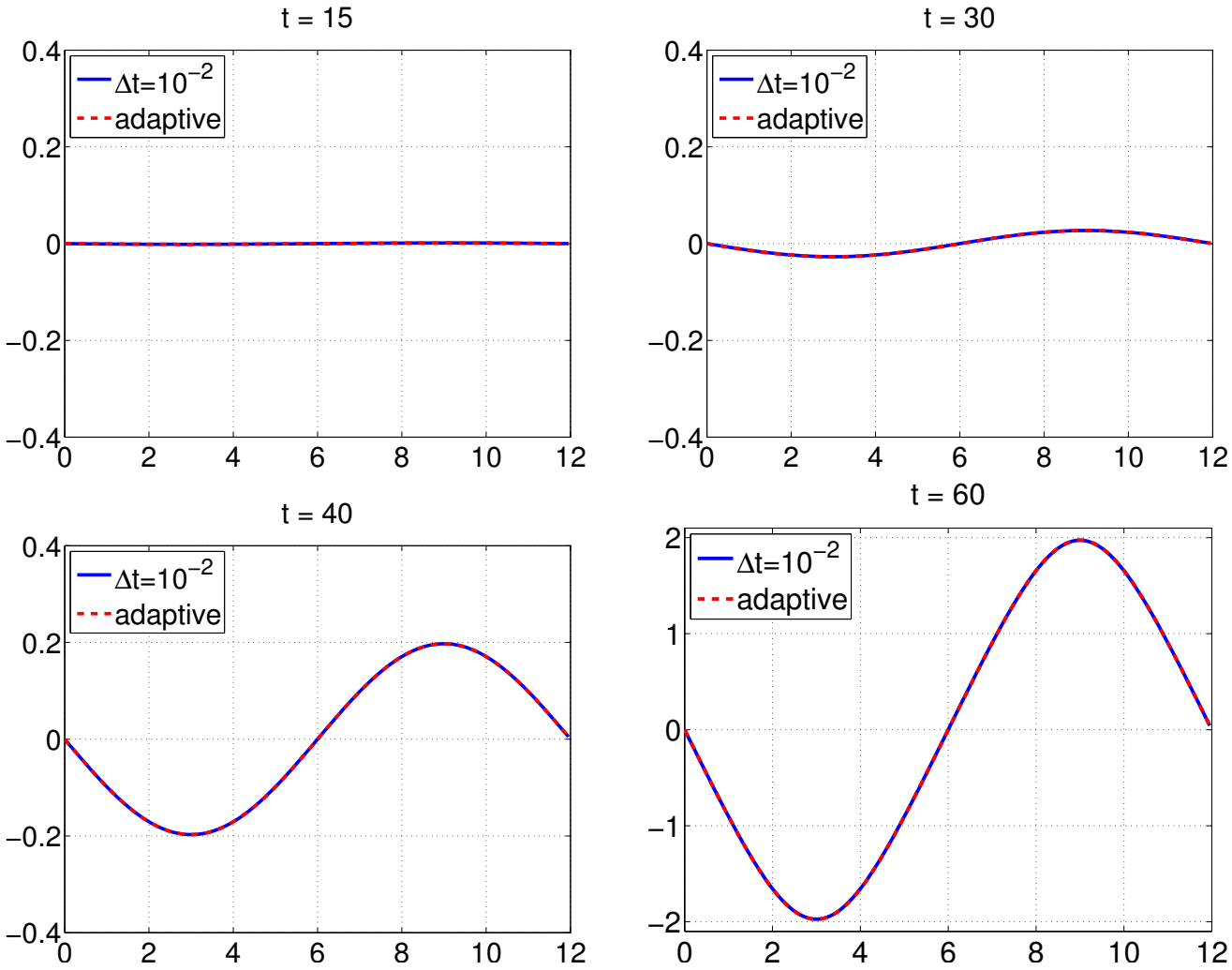


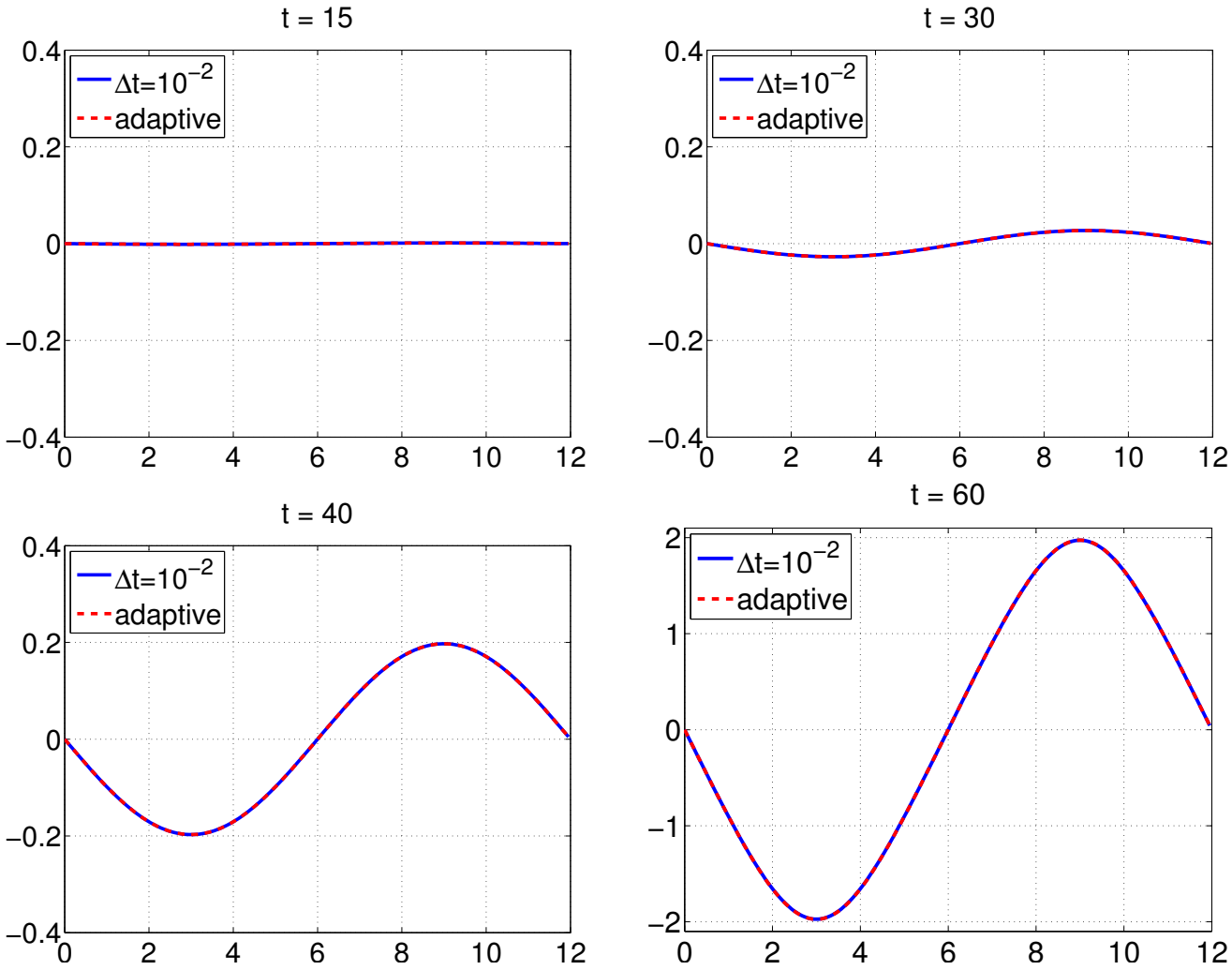
Figure: Left: [Li, Liu; 2003]; Right: our computation.

Difference in "buffering" time?

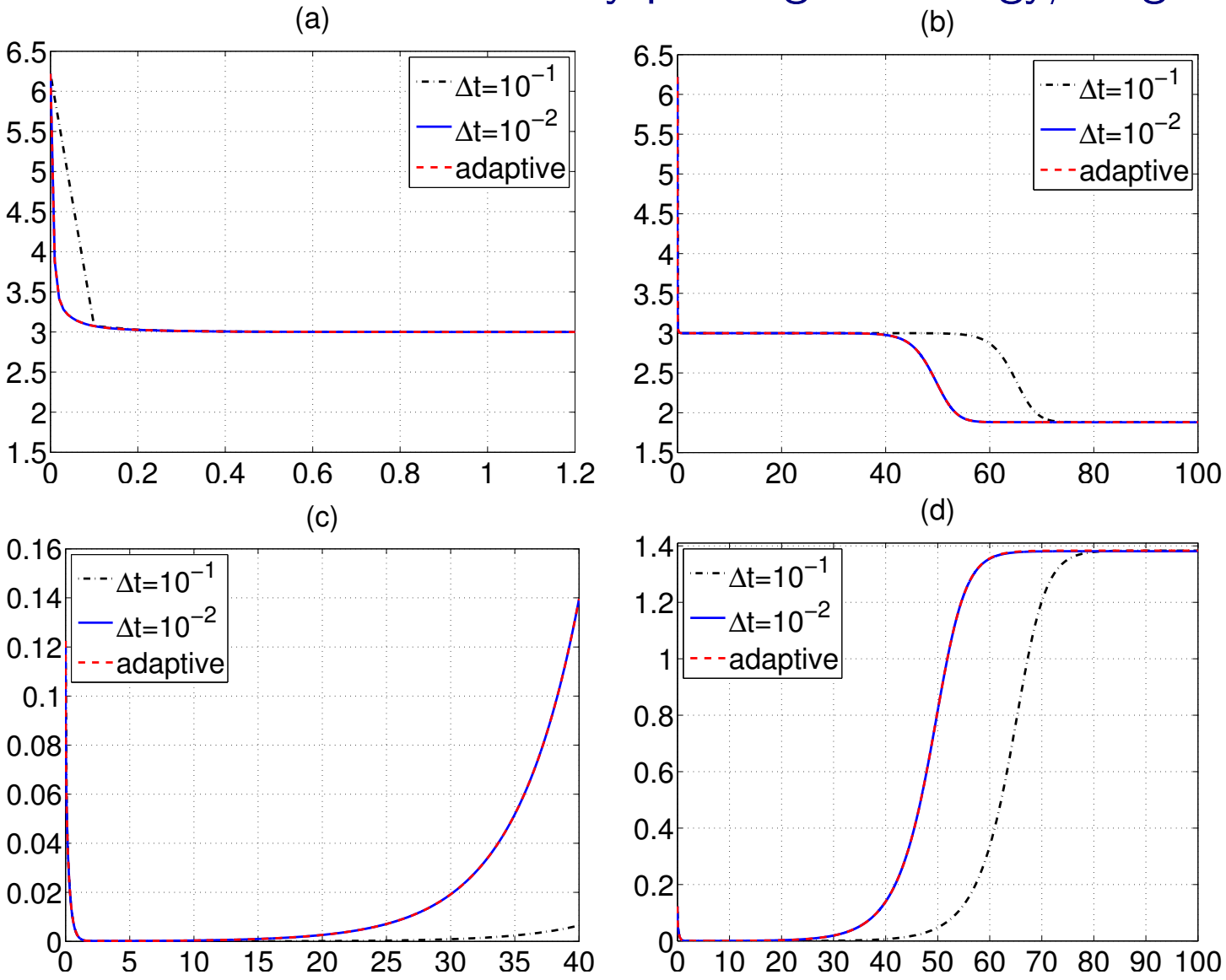
We therefore reduce the splitting step by a factor of 10 and repeat the computation with $\Delta t = 10^{-2}$. $\Delta t_{\max} = 10^{-1}$ $\Delta t_{\min} = 10^{-2}$ $\alpha = 10^{3}$



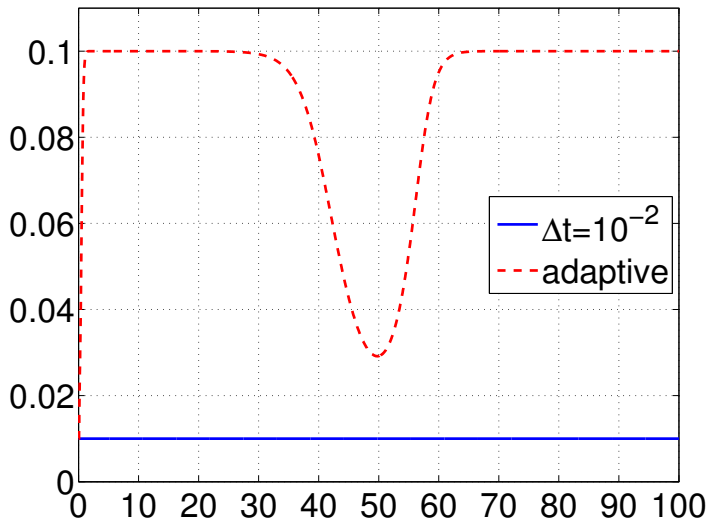
We therefore reduce the splitting step by a factor of 10 and repeat the computation with $\Delta t = 10^{-2}$. $\Delta t_{\max} = 10^{-1}$ $\Delta t_{\min} = 10^{-2}$ $\alpha = 10^3$



Monitor the time evolution process by plotting the energy/roughness:



It is instructive to check what splitting steps are used by the adaptive algorithm:



Example	N	T	Splitting step	CPU time
1	256	240	constant	3.2805
			adaptive	0.9659

Accuracy Test

Experimental convergence rate is close to the expected second-order one.

N	Δt	$\|\|u^{N,\Delta t} - u^{N/2,2\Delta t}\|\|_1$	Rate	$\|\|u^{N,\Delta t} - u^{N/2,2\Delta t}\|\|_\infty$	Rate
128	2e-2	3.95e-03	–	7.58e-04	–
256	1e-2	1.07e-03	1.89	2.45e-04	1.63
512	5e-3	2.73e-04	1.97	7.17e-05	1.78
1024	2.5e-3	6.84e-05	1.99	1.93e-05	1.89

Fix the splitting step to be $\Delta t = 10^{-3}$: the experimental convergence rate is fourth-order, which is the order of finite-difference scheme.

N	Δt	$\|\|u^{N,\Delta t} - u^{N/2,\Delta t}\|\|_1$	Rate	$\|\|u^{N,\Delta t} - u^{N/2,\Delta t}\|\|_\infty$	Rate
128	1e-3	8.06e-05	–	2.25e-05	–
256	1e-3	5.18e-06	3.96	1.44e-06	3.96
512	1e-3	3.27e-07	3.99	9.10e-08	3.99
1024	1e-3	2.02e-08	4.02	5.62e-09	4.02

Accuracy Test

Experimental convergence rate is close to the expected second-order one.

N	Δt	$\|\|u^{N,\Delta t} - u^{N/2,2\Delta t}\|\|_1$	Rate	$\|\|u^{N,\Delta t} - u^{N/2,2\Delta t}\|\|_\infty$	Rate
128	2e-2	3.95e-03	–	7.58e-04	–
256	1e-2	1.07e-03	1.89	2.45e-04	1.63
512	5e-3	2.73e-04	1.97	7.17e-05	1.78
1024	2.5e-3	6.84e-05	1.99	1.93e-05	1.89

Fix the splitting step to be $\Delta t = 10^{-3}$: the experimental convergence rate is fourth-order, which is the order of finite-difference scheme.

N	Δt	$\|\|u^{N,\Delta t} - u^{N/2,\Delta t}\|\|_1$	Rate	$\|\|u^{N,\Delta t} - u^{N/2,\Delta t}\|\|_\infty$	Rate
128	1e-3	8.06e-05	–	2.25e-05	–
256	1e-3	5.18e-06	3.96	1.44e-06	3.96
512	1e-3	3.27e-07	3.99	9.10e-08	3.99
1024	1e-3	2.02e-08	4.02	5.62e-09	4.02

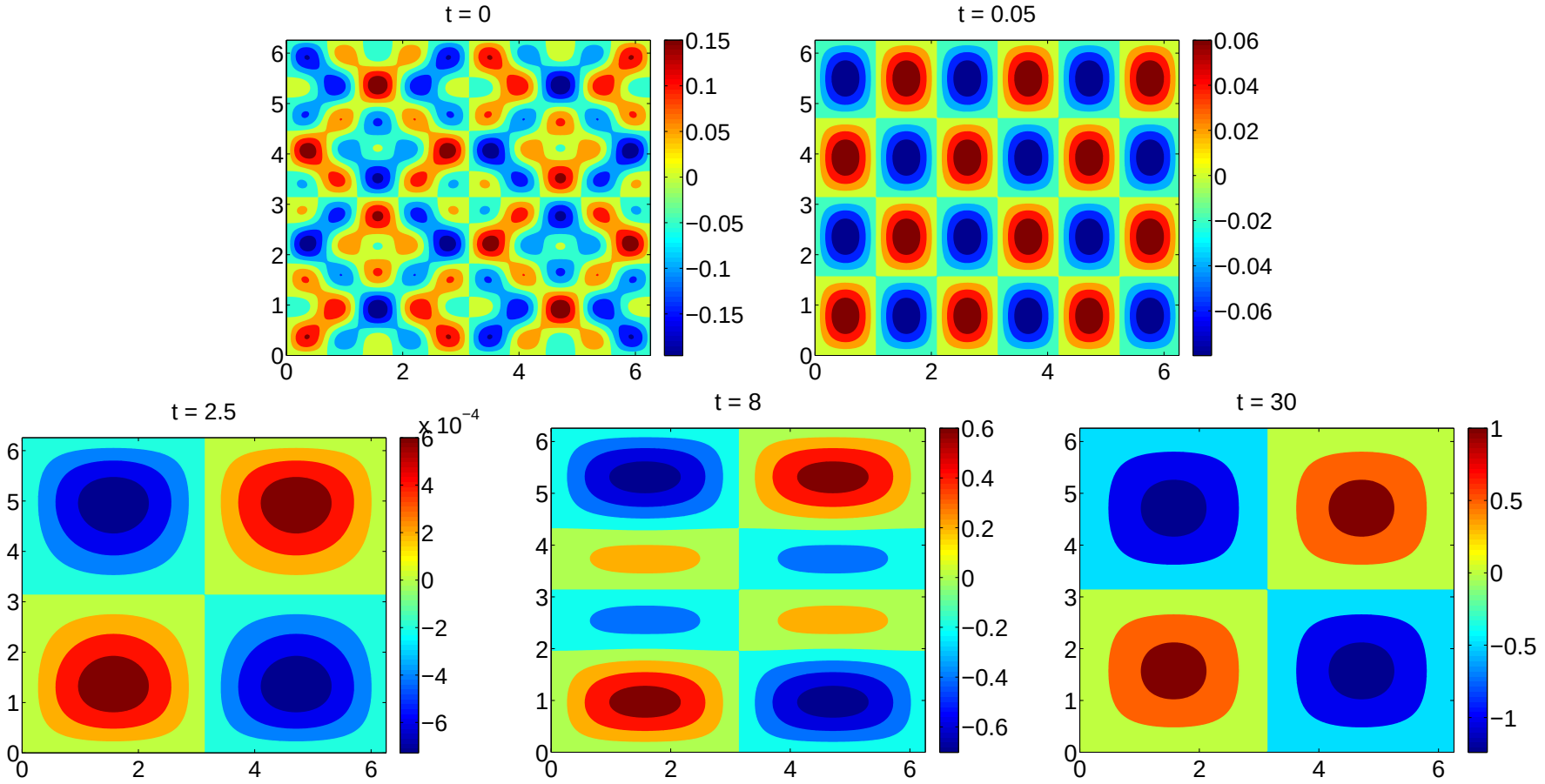
Example 2 — Two-Dimensional Morphological Instability

We consider the 2-D MBE equation with $\delta = 0.1$ subject to the following initial condition:

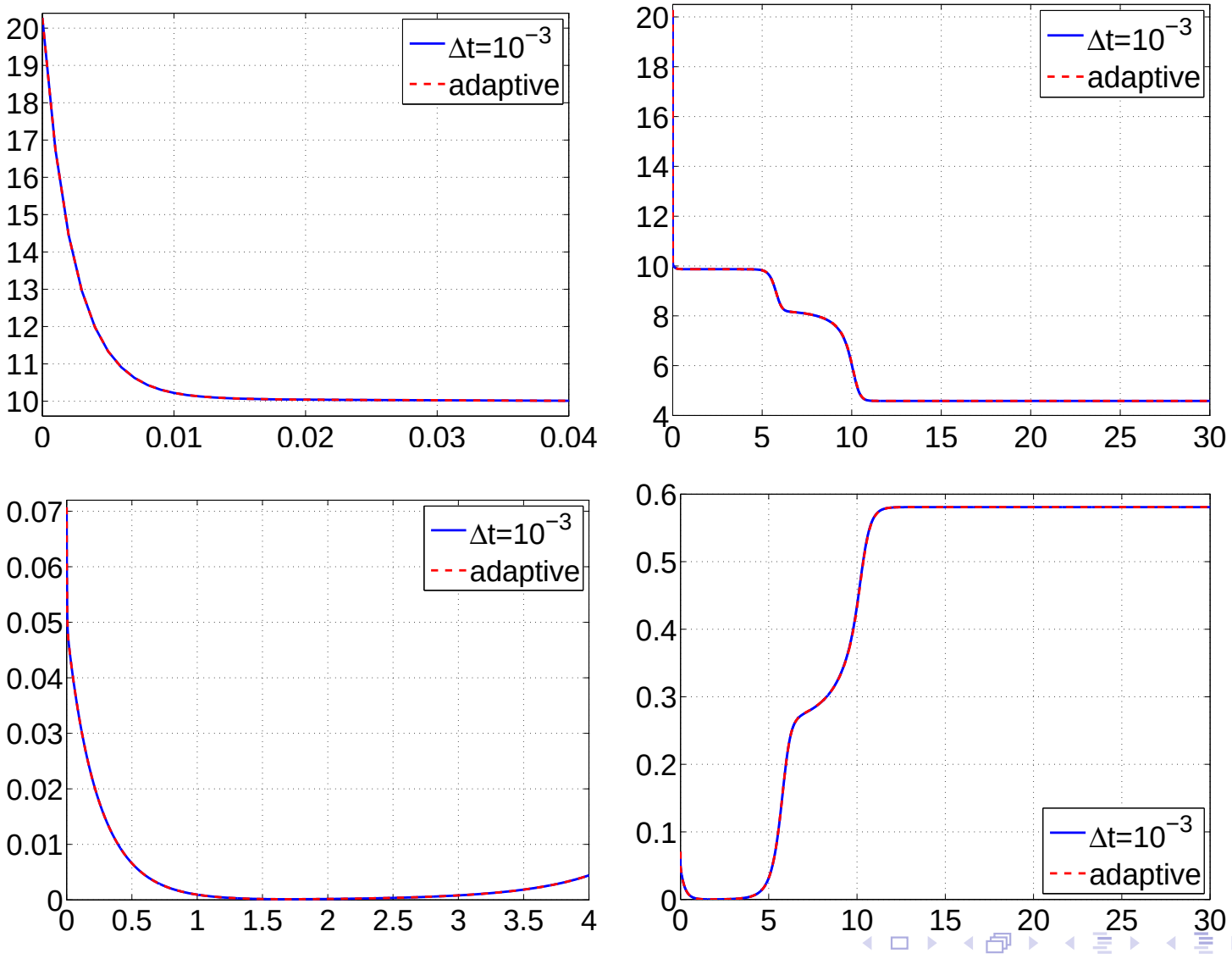
$$u(\boldsymbol{x}, 0) = 0.1(\sin 3x \sin 2y + \sin 5x \sin 5y), \quad \boldsymbol{x} \in [0, 2\pi]^2$$

This example was studied in [Li, Liu; 2003] [Xu, Tang; 2006] to observe the morphological instability due to the nonlinear interaction.

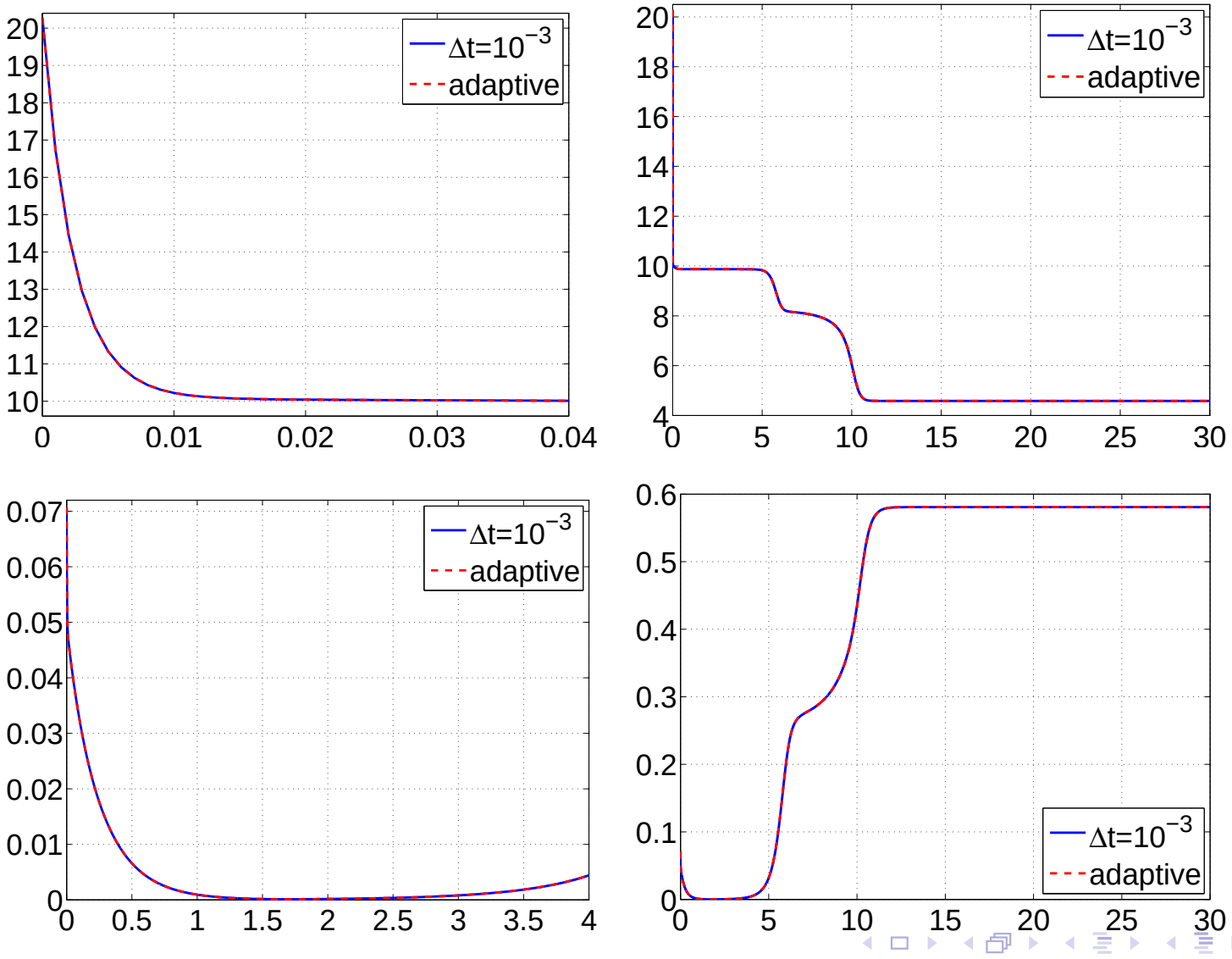
We compute the solution on a 256×256 uniform grid with the constant splitting step $\Delta t = 10^{-3}$. Contour plots of the height profiles:



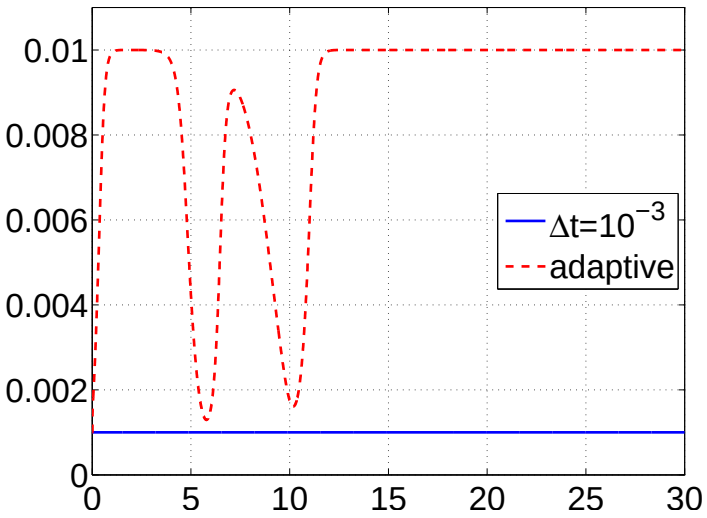
Experimental energy decay and roughness development. Consistent with [Li, Liu; 2003]; Adaptive strategy: $\Delta t_{\min} = 10^{-3}$ $\Delta t_{\max} = 10^{-2}$ $\alpha = 10^{3}$



Experimental energy decay and roughness development. Consistent with [Li, Liu; 2003]; Adaptive strategy: $\Delta t_{\min} = 10^{-3}$ $\Delta t_{\max} = 10^{-2}$ $\alpha = 10^3$



Splitting steps evolution shows that $\Delta t \approx \Delta t_{\max}$ when the solution approaches its steady state. This leads to a substantial decrease in CPU time.



Example	N	T	Splitting step	CPU time
2	256	30	constant	4601.9
			adaptive	838.9

Accuracy Test

Finally, we perform the mesh-refinement study and verify the experimental convergence rates are close to the expected second-order one.

N	Δt	$\|\|u^{N,\Delta t} - u^{N/2,2\Delta t}\|\|_1$	Rate	$\|\|u^{N,\Delta t} - u^{N/2,2\Delta t}\|\|_\infty$	Rate
64	4e-3	3.36e-03	–	6.01e-04	–
128	2e-3	9.09e-04	1.88	1.55e-04	1.96
256	1e-3	2.48e-04	1.87	4.96e-05	1.64
512	5e-4	6.52e-05	1.93	1.55e-05	1.68

Example 3 — Coarsening Dynamics

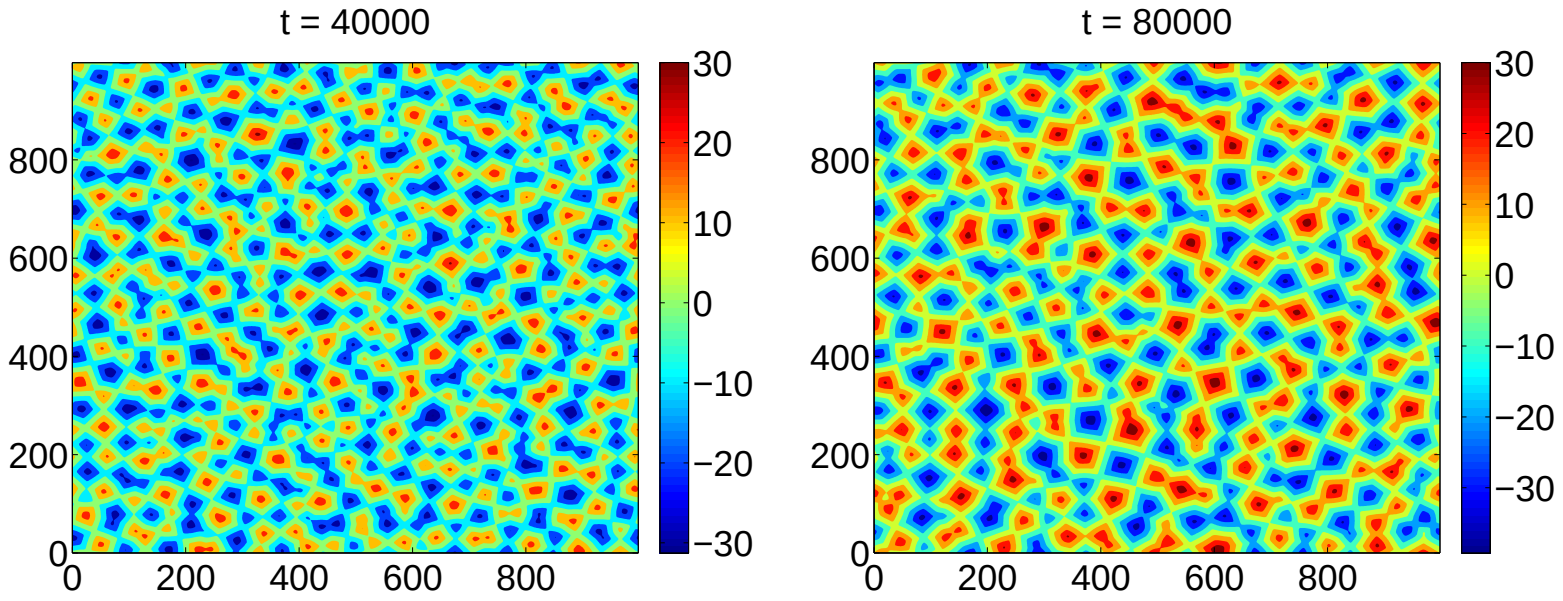
In this example, we study the 2-D MBE equation with $\delta = 1$ subject to random initial data:

- assign a uniformly distributed random number in the range $[-0.001, 0.001]$ to each grid point value of $u(\boldsymbol{x}, 0)$
- use a 512×512 uniform grid on the computational domain $\Omega = [0, 1000]^2$

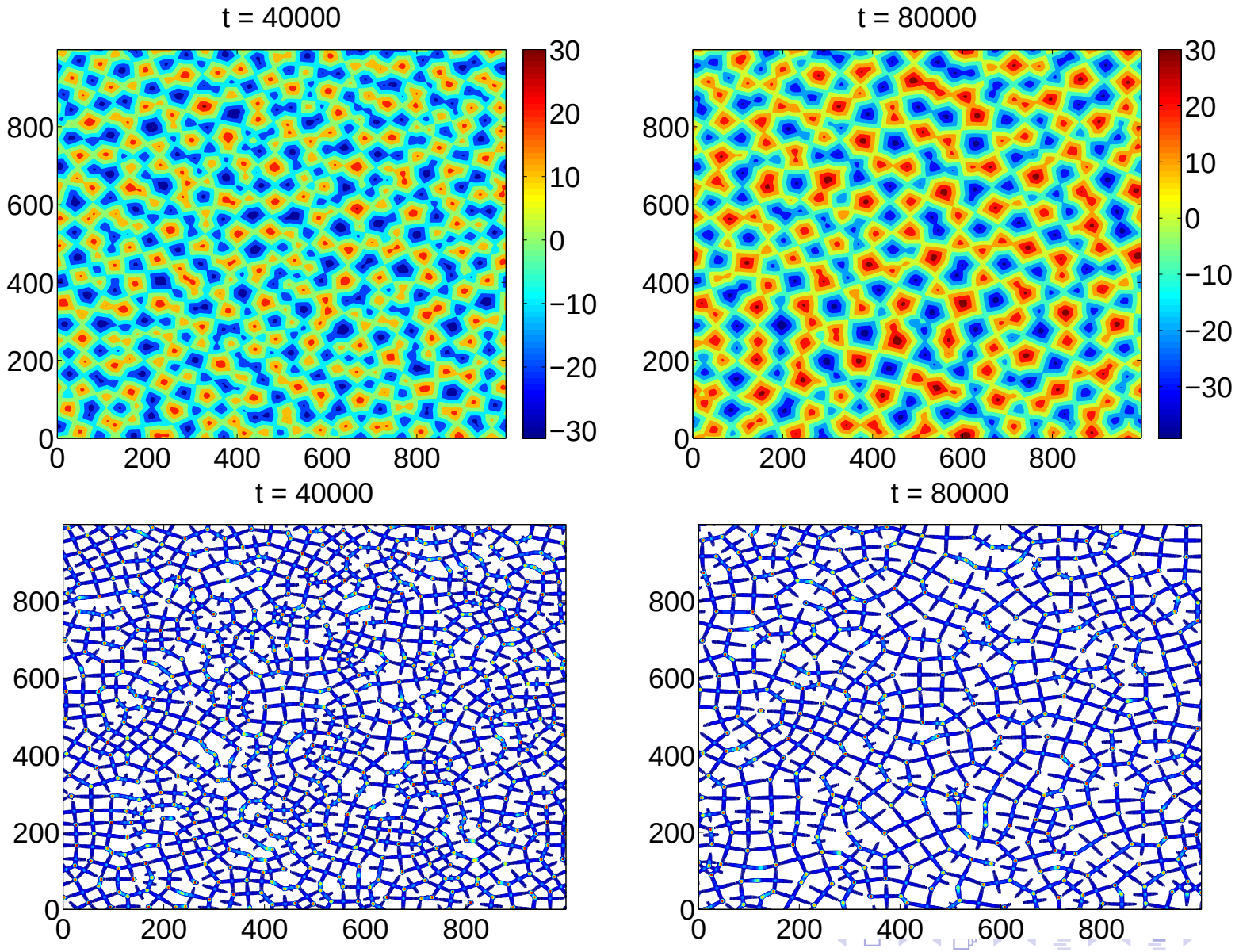
Free energy function

$$F_{free} := \frac{1}{4}(|\nabla u| - 1)^2 + \frac{\delta}{2}|\Delta u|^2$$

Contour lines of the solution profiles and free energy function ($\Delta t = 0.1$)



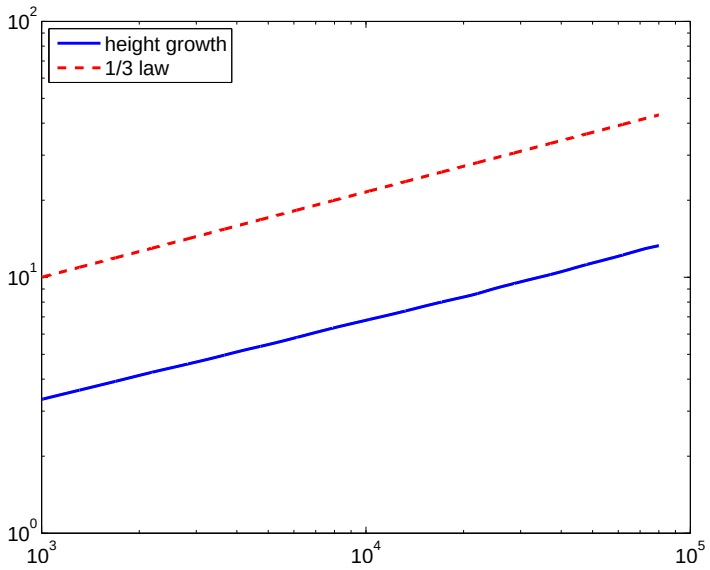
Contour lines of the solution profiles and free energy function ($\Delta t = 0.1$)

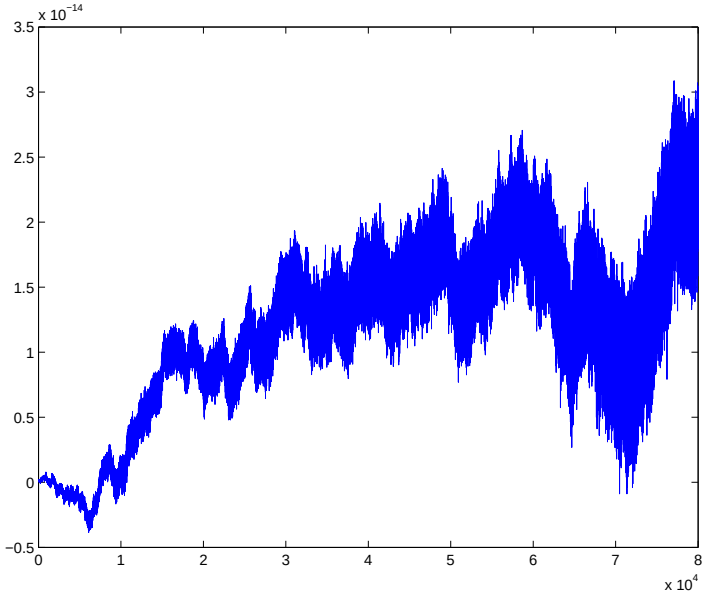


We present the log-log scale plot of the interface height (left):

$$\tilde{u}(t) = \left(\frac{1}{|\Omega|} \int_{\Omega} u^2(\boldsymbol{x}, t)\, d\boldsymbol{x} \right)^{\frac{1}{2}}$$

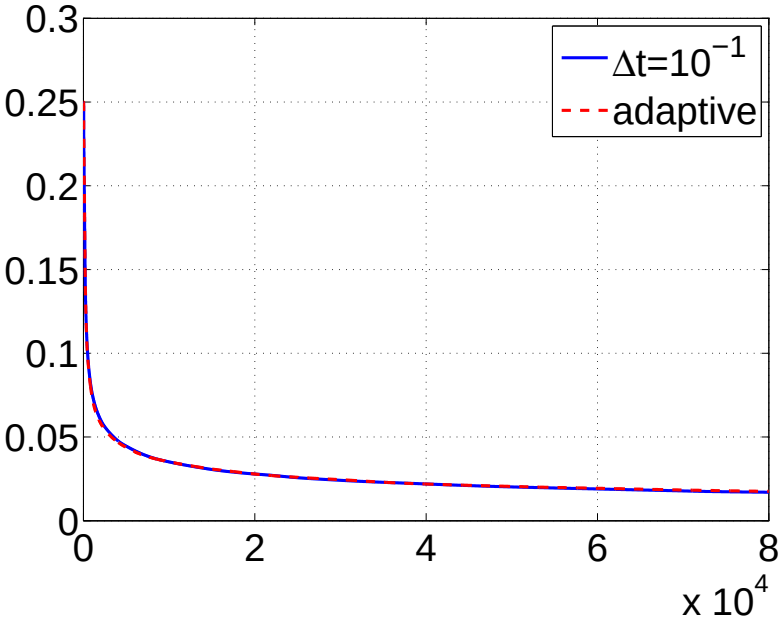
and the evolution of the mean height $\bar{u}(t)$ (right)

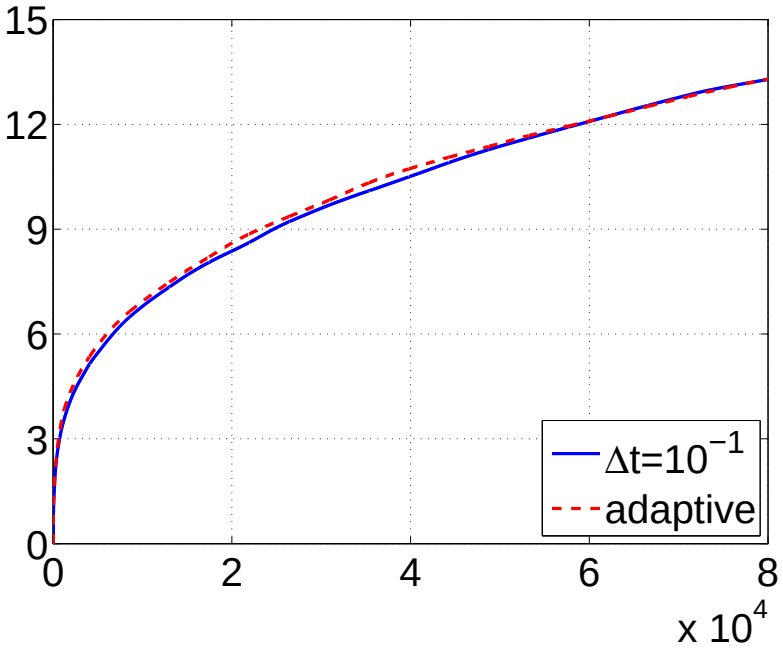




- The height of the pyramids grow in time as a power law $Ct^{1/3}$.
- The difference $\bar{u}(t) - \bar{u}(0)$ remains practically zero at all times $\Rightarrow$ mass conservation.

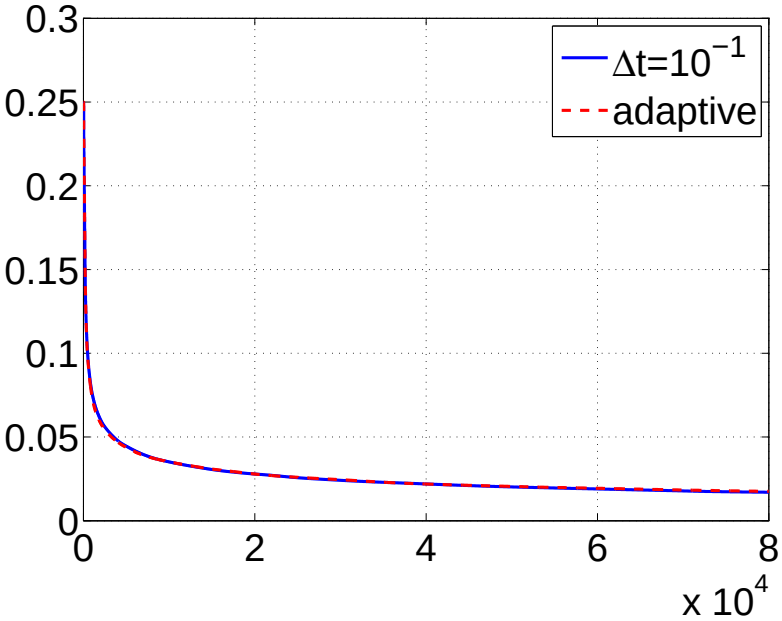
Energy, normalized by the domain size, and roughness development ($\Delta t = 10^{-1}$); Adaptive strategy: $\Delta t_{\min} = 10^{-1}$ $\Delta t_{\max} = 5$ $\alpha = 1$

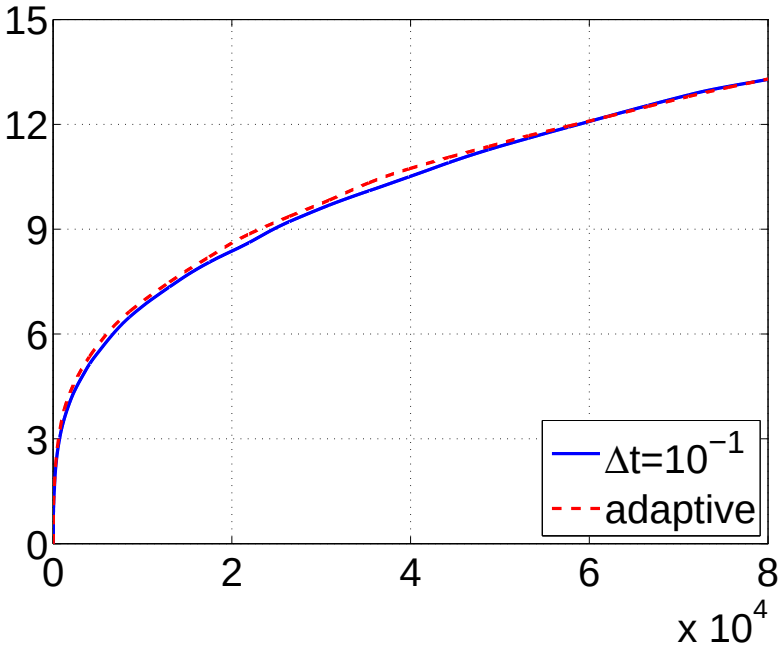




The obtained results match the experimental and numerical ones reported in [Moldovan, Golubovic; 2000] [Xu, Tang; 2006].

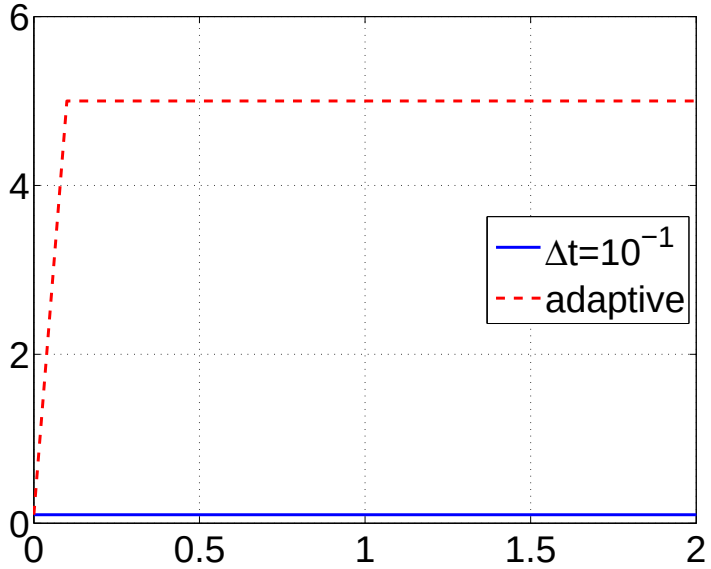
Energy, normalized by the domain size, and roughness development ($\Delta t = 10^{-1}$); Adaptive strategy: $\Delta t_{\min} = 10^{-1}$ $\Delta t_{\max} = 5$ $\alpha = 1$





The obtained results match the experimental and numerical ones reported in [Moldovan, Golubovic; 2000] [Xu, Tang; 2006].

Splitting step evolution



The splitting step increases to $\Delta t_{\max}$ very soon and then is always taken close to $\Delta t_{\max}$ due to the slow variation of the roughness. This leads to substantial CPU time usage saving:

Example	N	T	Splitting step	CPU time
3	512	80000	constant	223370
			adaptive	38775

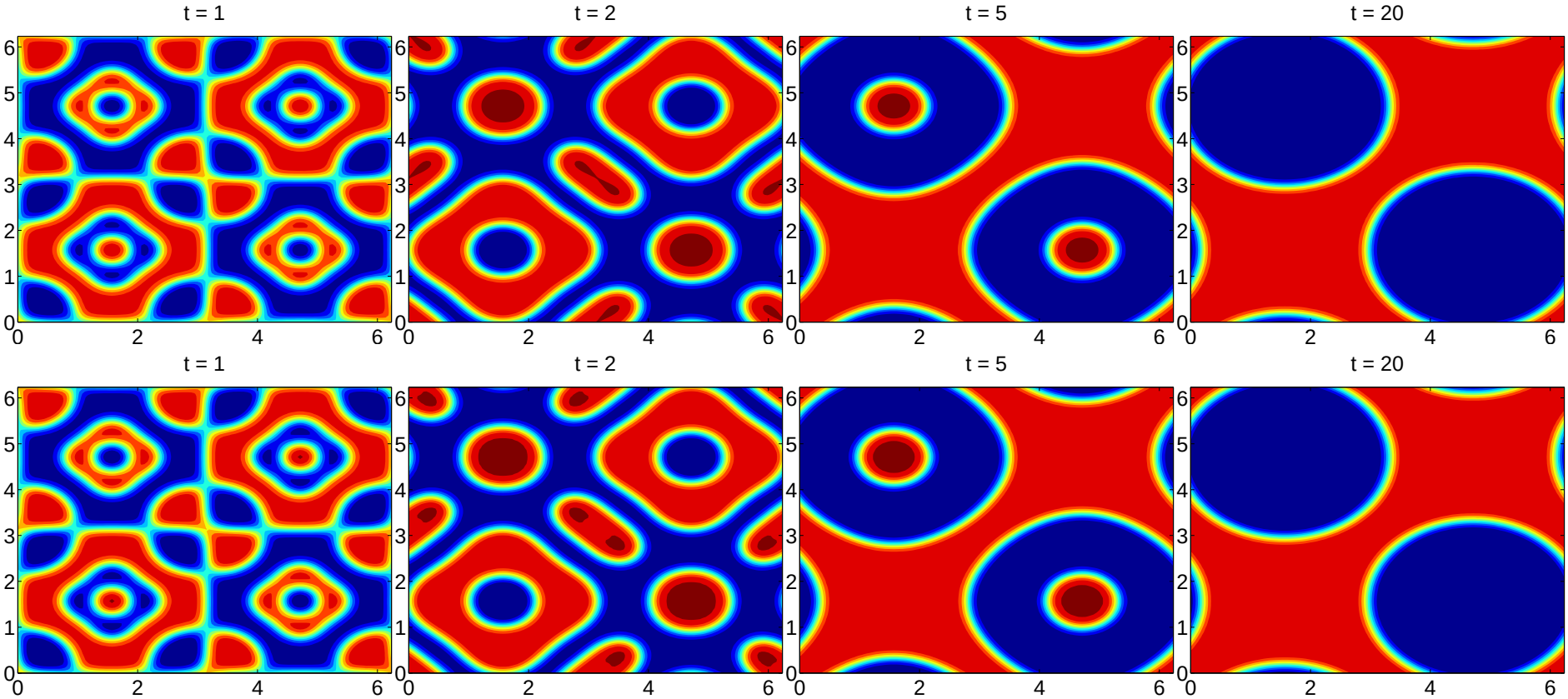
Example 4 — Phase Separation

In this example, taken from [Feng, Tang, Yang; 2015], we consider the 2-D CH equation with $\delta = 0.01$ subject to the following non-mean-zero initial condition:

$$u(\boldsymbol{x}, 0) = 0.05 \sin x \sin y + 0.001, \quad \boldsymbol{x} \in [0, 2\pi]^2$$

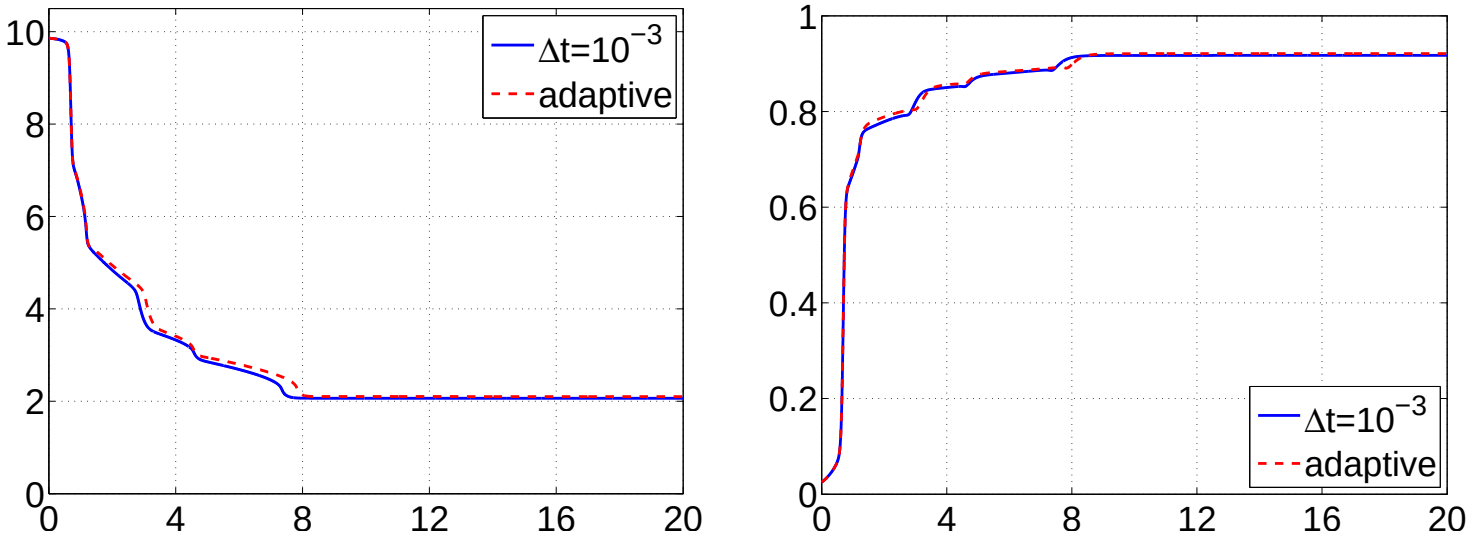
- 128×128 uniform grid
- constant splitting step $\Delta t = 10^{-3}$

Contour plots of solution computed with $\Delta t = 10^{-3}$ (upper row)adaptive strategy with $\Delta t_{\min} = 10^{-3}$ $\Delta t_{\max} = 10^{-2}$ $\alpha = 10^2$ (lower row).



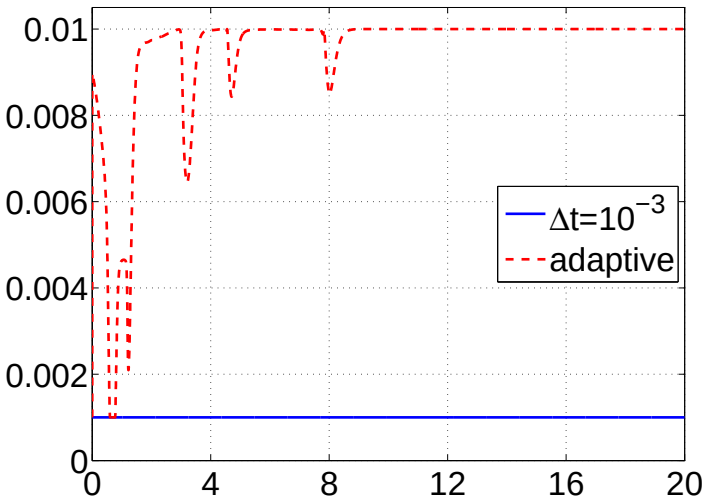
The solution dynamics can be captured correctly when the adaptive strategy is employed.

Energy and roughness curves



The curves have some discrepancy with those obtained using the small constant splitting step Δt^{-3}, though the adaptive and non-adaptive solutions are quite close and the resulting steady states seem to be the same.

Splitting steps evolution



$\Delta t \approx \Delta t_{\max}$ when the solution approaches its steady state, which leads to a substantial saving in CPU time:

Example	N	T	Splitting step	CPU time
4	128	20	constant	504.09
			adaptive	125.86

Outlines

Thanks for your attention.