Adaptive Finite Difference Method with Variable Timesteps for Fractional Diffusion and Diffusion-Wave Problems

Santos B. Yuste and Joaquín Quintana-Murillo

Dpt. Física, Univ. Extremadura

Badajoz (Spain)

JUNTA DE EXTREMADURA
Vicepresidencia Segunda y Consejería de Economía, Comercio e Innovación

UNIÓN EUROPEA
Fondo Europeo de Desarrollo Regional
Una manera de hacer Europa

SPhinX
Statistical Physics in eXtremadura

The problem in a nutshell

For **some anomalous diffusive (subdiffusive) particles**, the pdf of finding them at a given place at a given ***time*** follows **a fractional (i.e., an integro-differential) diffusion equation**. When one solves this equation by means of standard finite difference methods, the CPU time and computer memory consumption scale as $time^2$ **!!!**

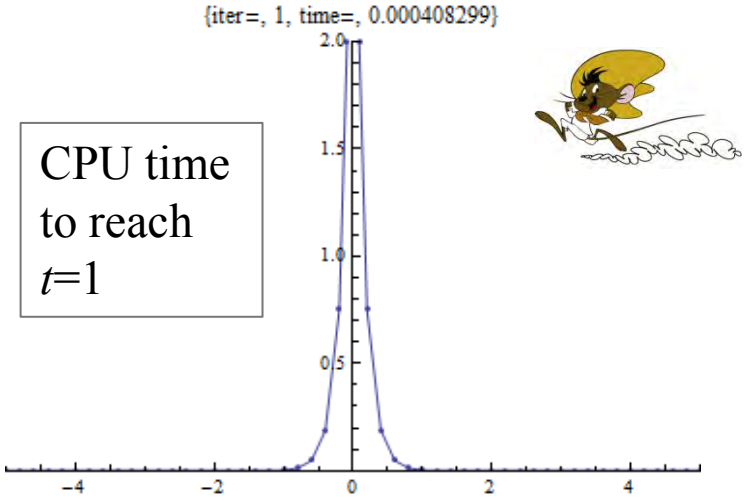
Solution from t= 0 to t =**1**

CPU time to reach t=1

>>

Solution from t= 0 to t =**2**

CPU time to reach t=1

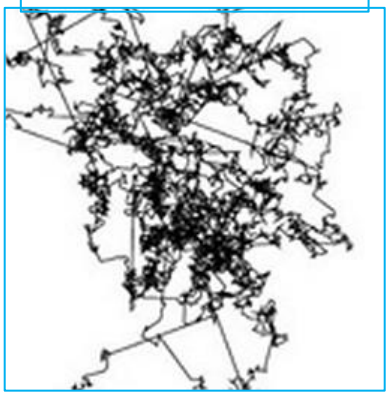


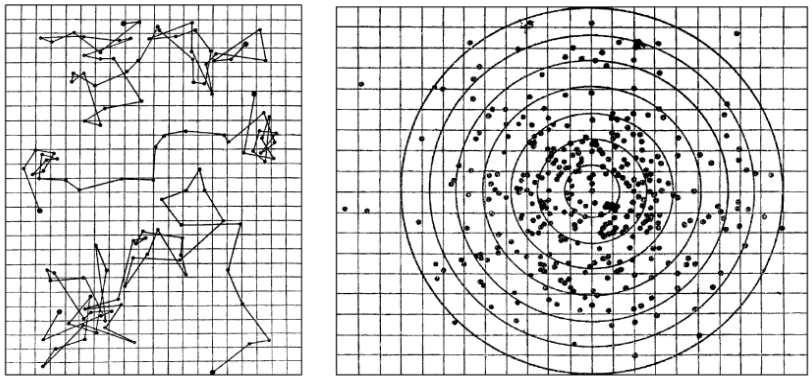
Standard method

Our adaptive method

Brownian motion. Normal diffusion

Fig. from the web page of this Symposium



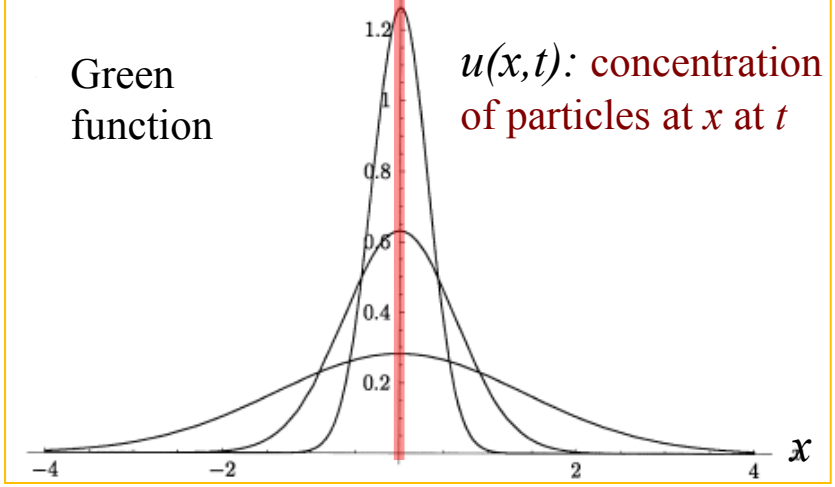


Recorded random walk trajectories by Jean Baptiste Perrin. Lef part: three designs obtained by tracing a small grain at intervals of 30 s. Right part: the starting point of each motion event is shifted to the origin. These figures constitute part of the measurement of Perrin, Dabrowski and Chaudesaigues leading to the determination of the Avogadro number. The result was 7.05×10^{23}

R. Metzler, J. Klafter / Physics Reports 339 (2000) 1-77

Jean Baptiste Perrin

Green function



u(x,t): concentration of particles at *x* at *t*

$$\langle x^2 \rangle \sim t$$

$$\frac{\partial}{\partial t}u(x,t) = D\frac{\partial^2}{\partial x^2}u(x,t)$$

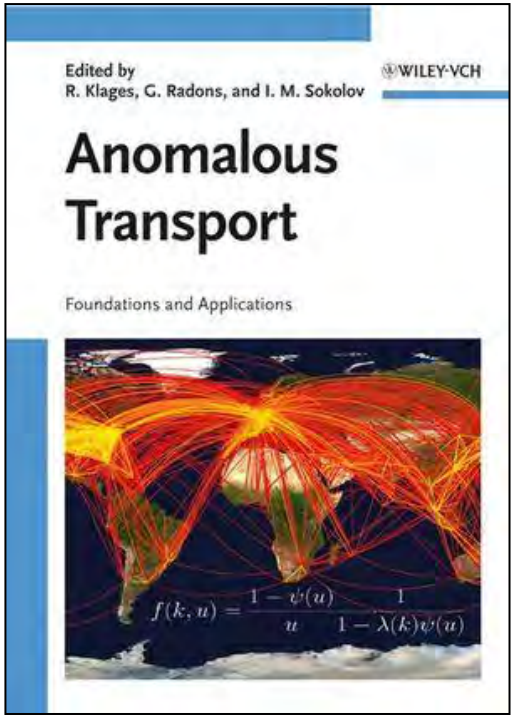
Diffusion equation

$$u(x,t) = \frac{1}{\sqrt{4\pi Dt}}e^{-x^2/4Dt}$$

Green function

Subdiffusion in...

- Physics • Geology • Finance
- Chemistry • Ecology • …….
- *Biology* (see next)



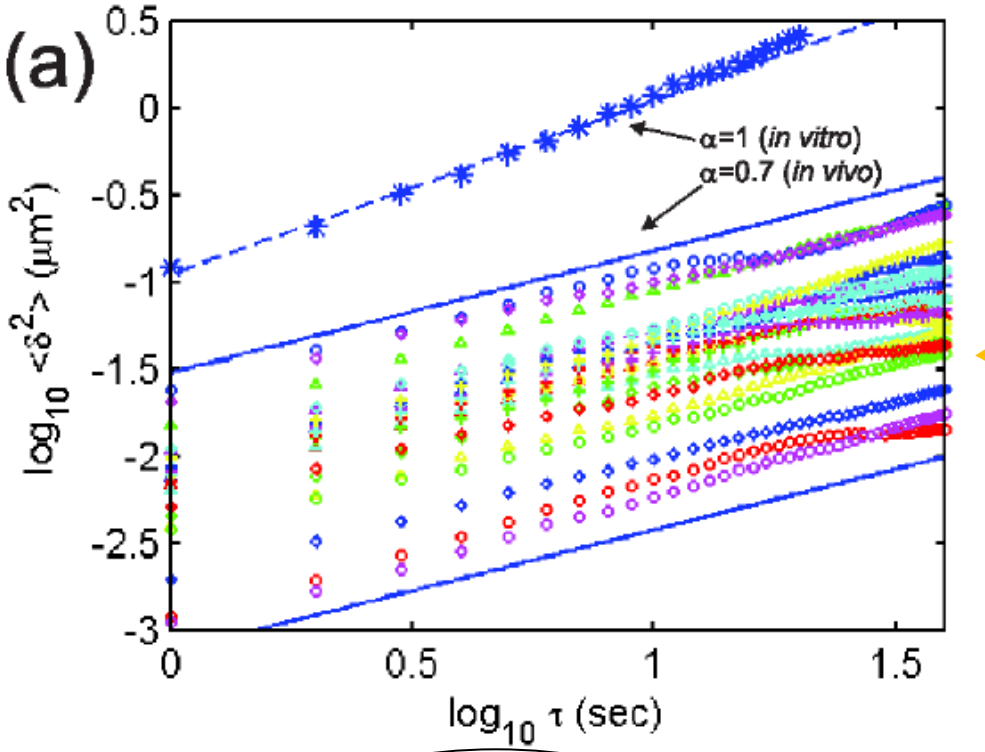
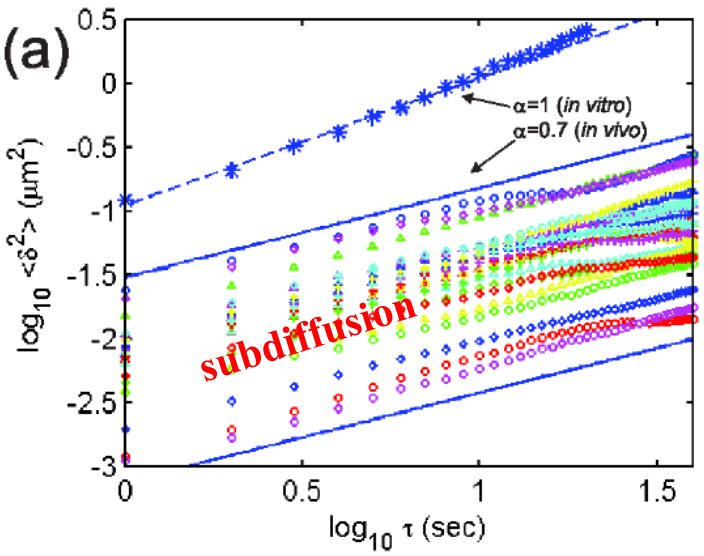


FIG. 2 (color online). Subdiffusive motion of RNA molecules in the cell. Movies were read into Matlab software (Mathworks).

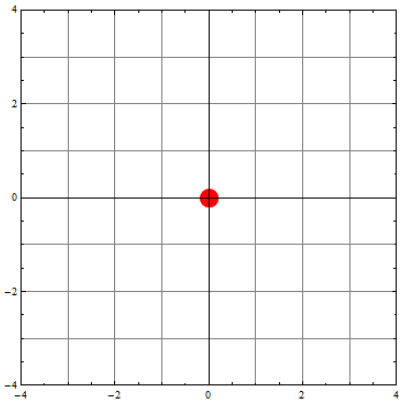
(anomalous) diffusion exponent *in vivo*: γ =0.7

Golding and Cox
Physical Nature of Bacterial Cytoplasm
PRL **96**, 098102 (2006)

How can we model subdiffusion processes?



CTRW with fat/heavy tail ⟶ **Subdiffusion**



CTRW

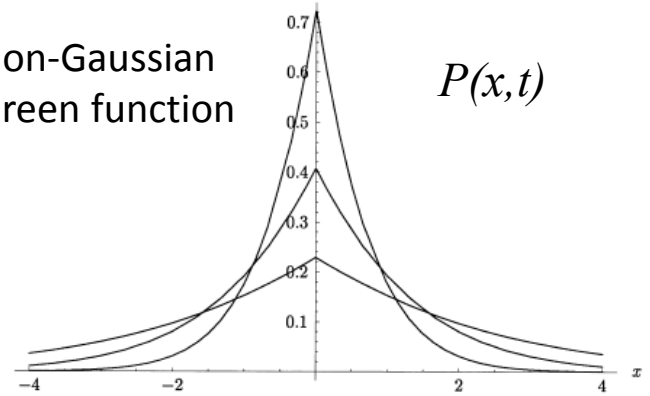
Jump distribution $\varphi(x)$ with finite variance and ***waiting-time distribution $\psi(t)$*** with power-law decay

$$\psi(t) \sim t^{-1-\gamma}$$

$$0 < \gamma < 1, \quad t \gg 1$$

GLCT ⟶

Non-Gaussian Green function



$$\langle r^2 \rangle \sim t^{\gamma}$$

Subdiffusion: $0<\gamma<1$

$$\frac{\partial^{\gamma}}{\partial t^{\gamma}} u(\vec{r},t) = K\nabla^2 u(\vec{r},t)$$

⇕

Fractional diffusion equation

Subdiffusion → Fractional diffusion equation

$$\langle x^2 \rangle \sim t^\gamma$$

$0<\gamma<1$

$$\frac{\partial^\gamma}{\partial t^\gamma} u(x,t) = K \frac{\partial^2}{\partial x^2} u(x,t)$$

Caputo derivative:

$$\frac{\partial^\gamma}{\partial t^\gamma} y(t) \equiv \frac{1}{\Gamma(1-\gamma)} \int_0^t d\tau \frac{1}{(t-\tau)^\gamma} \frac{dy(\tau)}{d\tau}, \quad 0 < \gamma < 1$$

- ☞ **Limit equation of** a (mesoscopic) **CTRW model** with power-law distribution of waiting times
- Linear
- **Easy inclusion of external fields and boundary conditions**
- ∃ **analytical techniques** and solutions for some basic problems (eigenfunction expansions, Green functions)
- ∃ algorithms for obtaining **numerical solutions**

Finite difference method: numerical scheme

$$\partial u = F \quad \rightarrow \quad \delta U = F$$

$$\left[\frac{\partial^\gamma}{\partial t^\gamma} - K\frac{\partial^2}{\partial x^2}\right] u(x,t) = F(x,t) \quad \rightarrow \quad \left[\delta_t^\gamma - K\delta_x^2\right] U_j^n = F(x_j,t_n) \qquad (1)$$

$$\delta_t^\gamma U_j^n = \frac{1}{\Gamma(2-\gamma)} \sum_{m=0}^{n-1} T_{m,n}^{(\gamma)} \left[U_j^{m+1} - U_j^m\right] \qquad (2)$$

$$\delta_x^2 u(x_j,t) = \frac{u(x_{j+1},t) - 2u(x_j,t) + u(x_{j-1},t)}{(\Delta x)^2} \qquad (3)$$

$$-S_n U_{j+1}^n + (1+2S_n)U_j^n - S_n U_{j-1}^n = \mathcal{M}U_j^n + \tilde{F}(x_j,t_n)$$

$$\mathcal{M}U_j^n \equiv U_j^{n-1} - \sum_{m=0}^{n-2} \tilde{T}_{m,n}^{(\gamma)} \left[U_j^{m+1} - U_j^m\right]$$

$$S_n = \Gamma(2-\gamma)K\frac{(t_n - t_{n-1})^\gamma}{(\Delta x)^2},$$

Fractional difference methods are *heavy*

t_0 t_{m-2} t_{m-1} t_m } → t_{m+1}

$$\left.\begin{array}{lll} U_{j-1}^0, & U_{j-1}^{m-2}, U_{j-1}^{m-1}, U_{j-1}^m \\ U_j^0, & U_j^{m-2}, U_j^{m-1}, U_j^m \\ U_{j+1}^0, & U_{j+1}^{m-2}, U_{j+1}^{m-1}, U_{j+1}^m \end{array}\right\} \longrightarrow \begin{bmatrix} U_{j-1}^{m+1} \\ U_j^{m+1} \\ U_{j+1}^{m+1} \end{bmatrix}$$

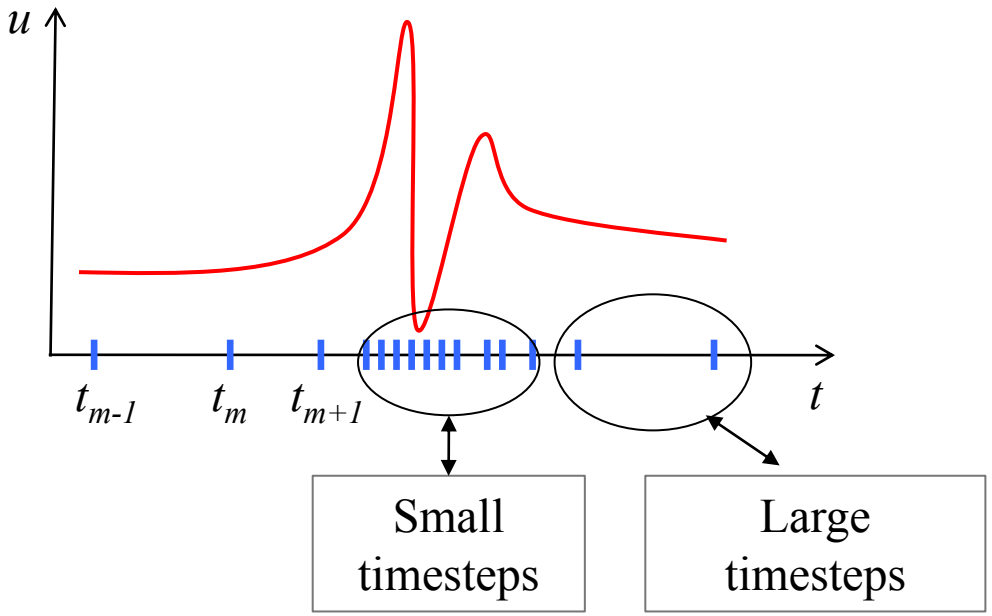
← m terms →

From m to m+1: computational cost ~ m

From m=*1* to m=n+*1*: computational cost $\sum_{m=1}^{n} m^2 \sim n^2$ Huge!!!

[computational cost for normal diffusion ~ n]

Variable/adaptive timesteps: Adaptive methods



Adaptive methods

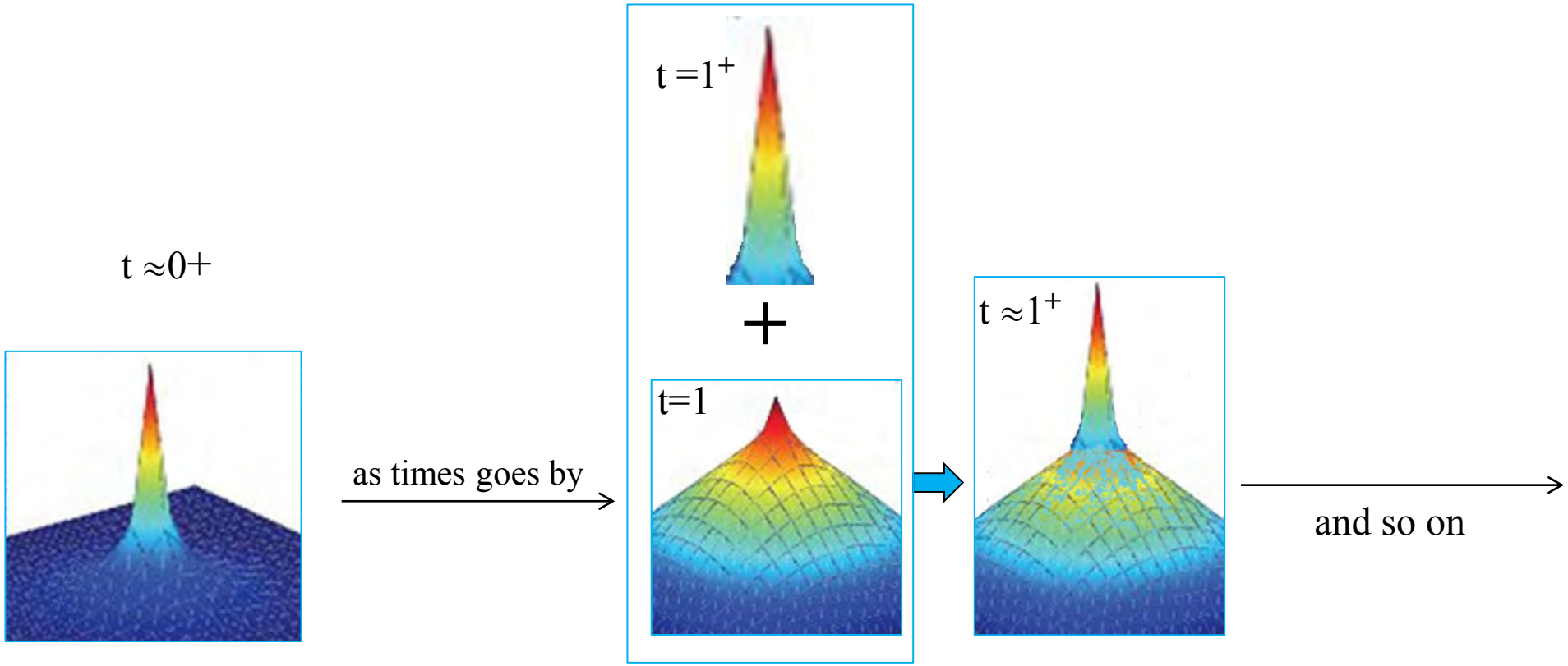
- ***more reliable***: via thorough sampling of difficult regions
- ***faster*** : via sparse sampling of quiet regions

A good ***ODE*** integrator should exert some adaptive control over its own progress, making frequent changes in its stepsize. […] Many small steps should **tiptoe** through treacherous terrain, while a few **great strides** should speed through smooth uninteresting countryside. The resulting gains in efficiency are not mere tens of percents or factors of two; they can sometimes be factors of ten, a hundred, or more

Press et al, *Numerical Recipes*, section 16.2

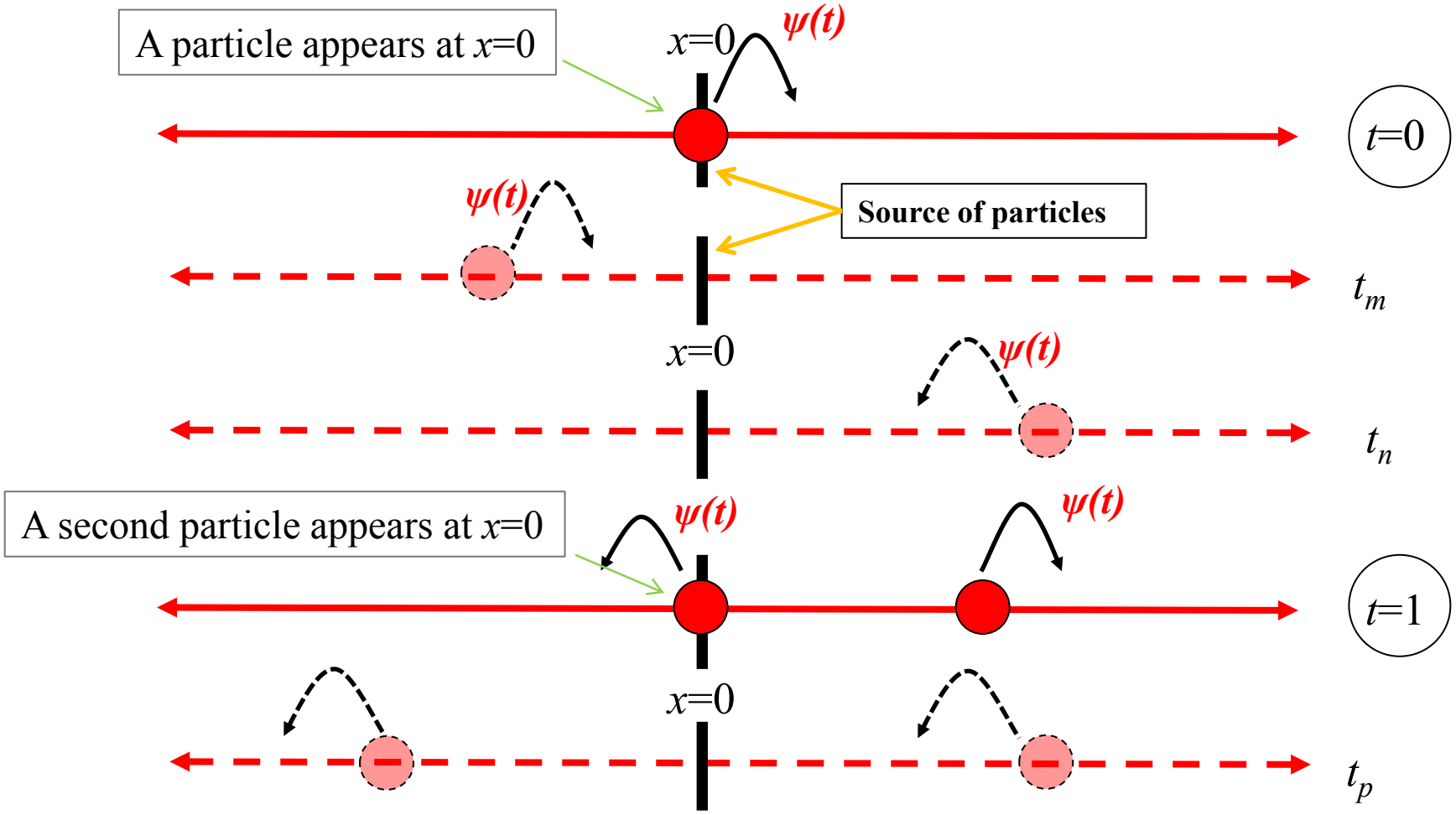
A testbed problem with *wild* and *quiet* regions

- Dispersion of a constant flux of subdiffusive particles arising from a point source
  (→ one particle released every unit time at the source)



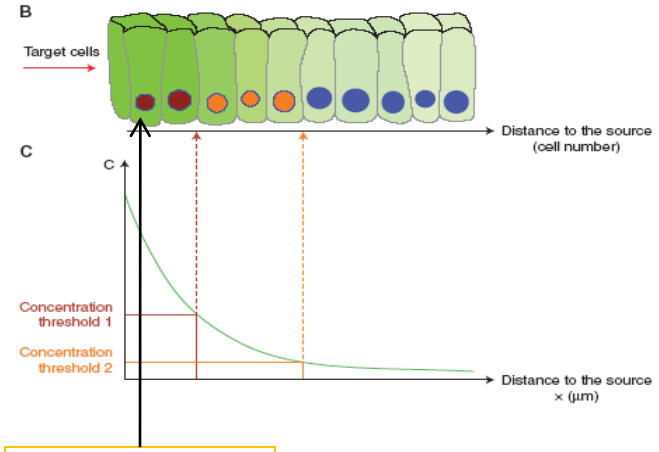
A testbed problem with *wild* and *quiet* regions

● Dispersion of a constant flux of subdiffusive particles ● one particle released every unit time at $x=0$:● **1D infinite medium**



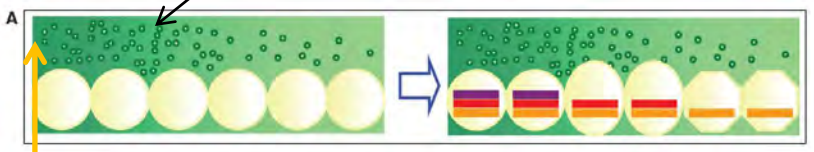
The goal: $u(x,t)$ = density of probability of finding a particle at x at time t

Formation of morphogen gradients: standard *reaction-diffusion* model



Source of particles

diffusion ↔ random walk



Source of particles

1D testbed problem

+

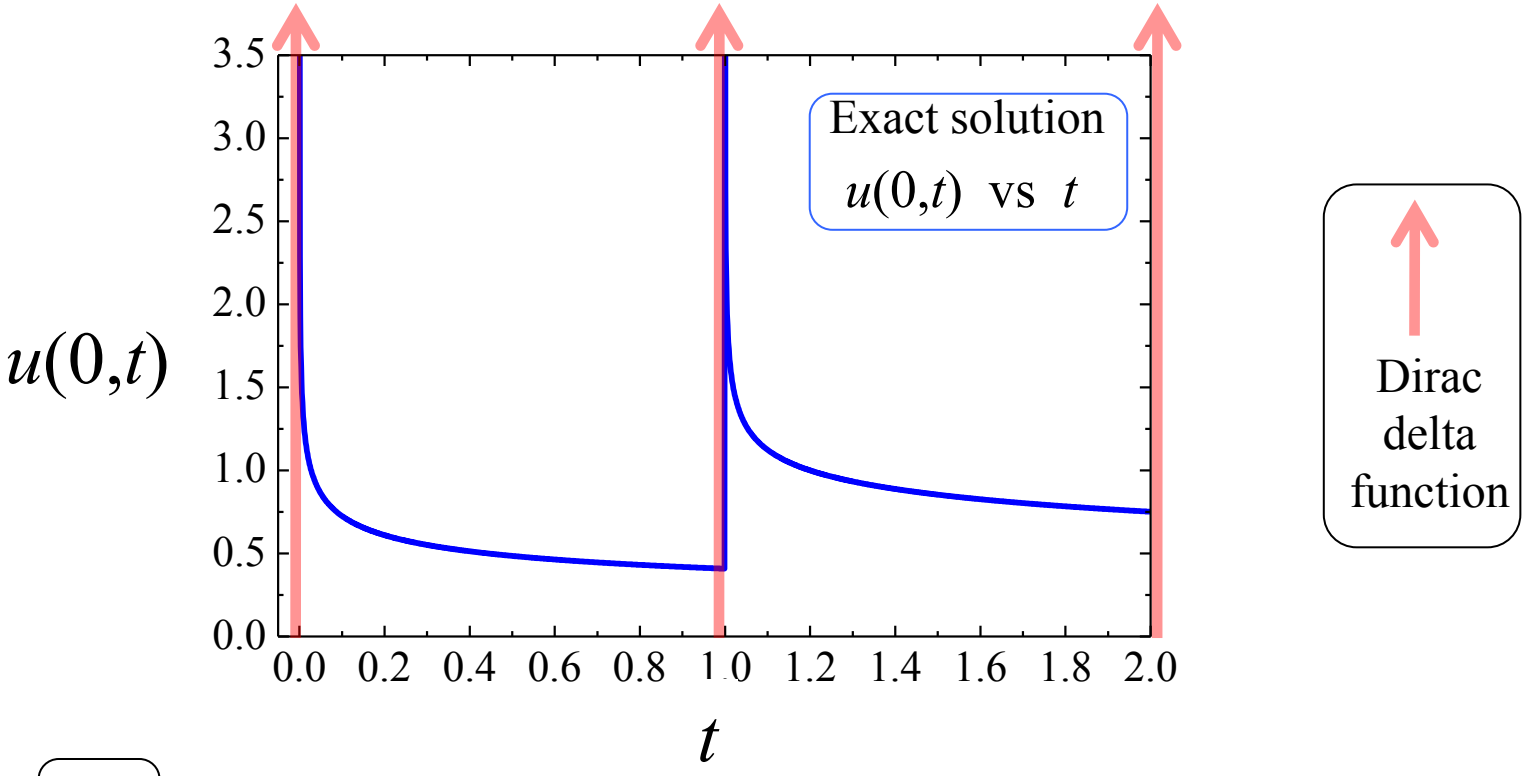
Particles (morphogens) may react and disappear

⇨ development of morphogens gradients

Long-time (stationary) distribution of morphogens in the medium is especially important ⇨ Fast and accurate numerical methods are required

A testbed problem with *wild* and *quiet* regions

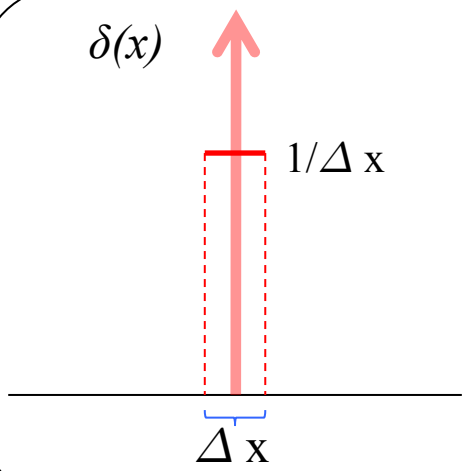
The release of a particle at $x=0$ is described by means of a Dirac delta function



n $\Longrightarrow$ a new particle appears at $t=n$: $u(x,n^+)=u(x,n^-)+\delta(x)$

Discretization of $\delta(x)$

Dirac delta function approximated by a hat function

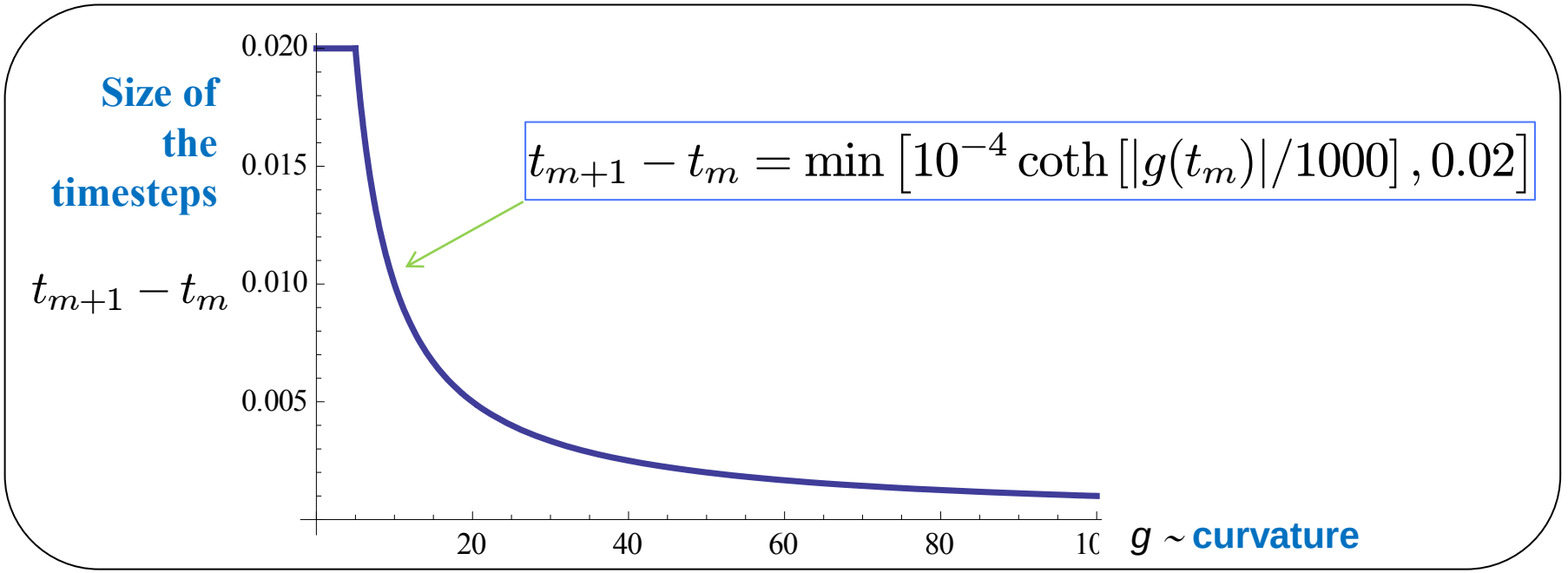


Discretized numerical approximation to the Dirac's delta initial condition:

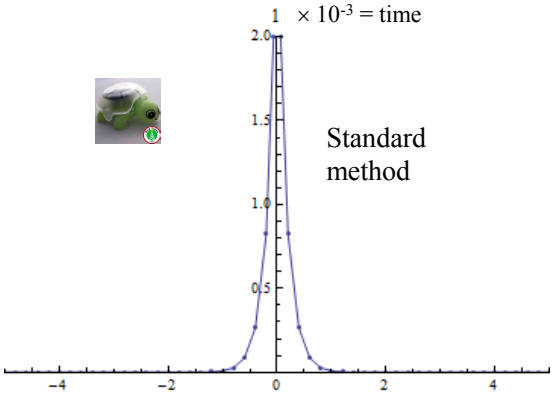
$$u(x_j, 0) = \begin{cases} 1/\Delta x, & j = 0 \\ 0, & j \neq 0 \end{cases}$$

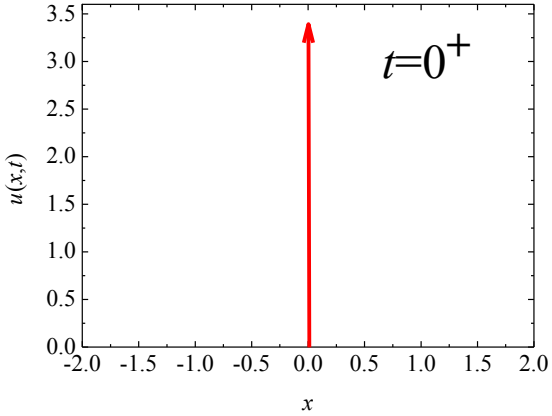
Choosing the timesteps

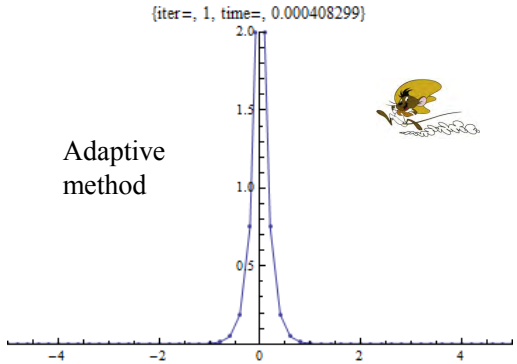
$g(t_m) = (U_{-1}^m - 2U_0^m + U_1^m)/(\Delta x)^2 \simeq \partial^2 u(x,t_m)/\partial x^2|_{x=0}$ ~ curvature at the origin

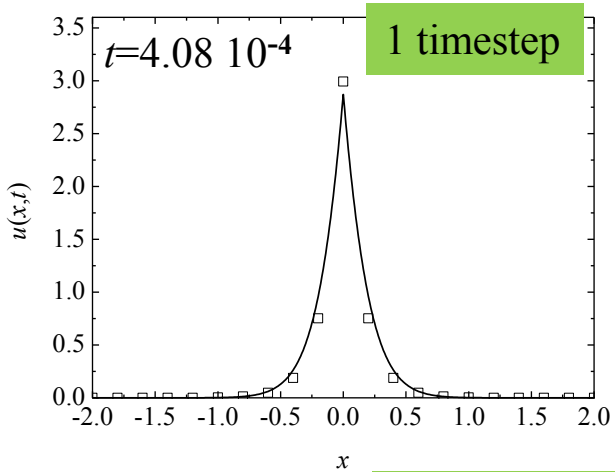


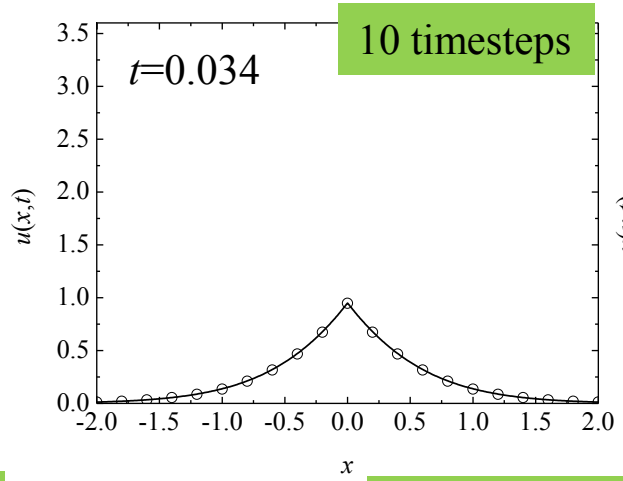
- *Arbitrary* but with convenient features:
  - Timestep decreases (increases) when the curvature increases (decreases)
  - Prefixed maximum and minimum values (0.02 and 0.0001, respectively)

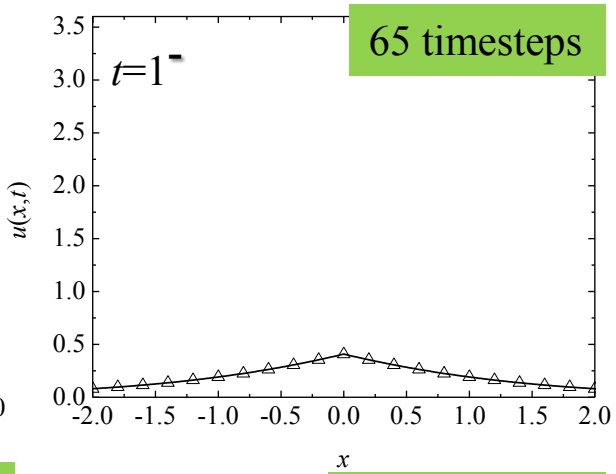

1 × 10-3 = time
Standard
method

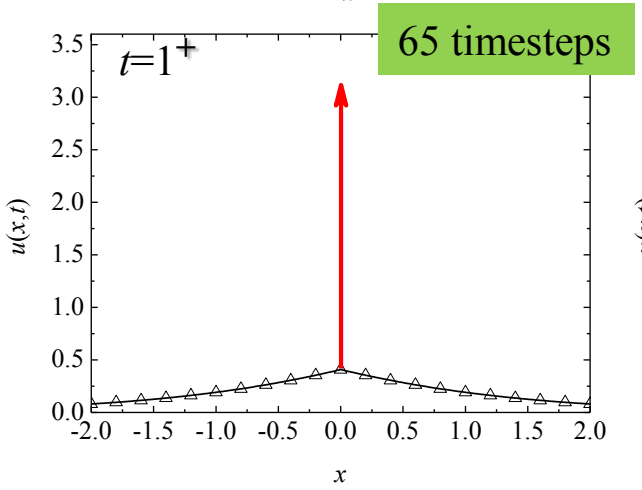

t=0+
u(x,t)
x

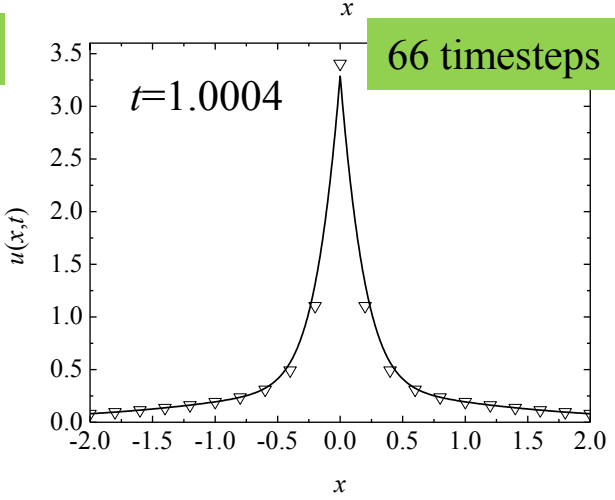

{iter=, 1, time=, 0.000408299}
Adaptive
method

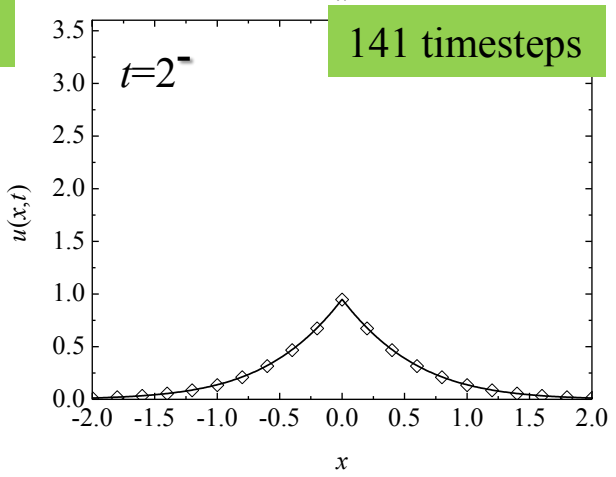

t=4.08 10-4
1 timestep
u(x,t)
x


t=0.034
10 timestep
u(x,t)
x

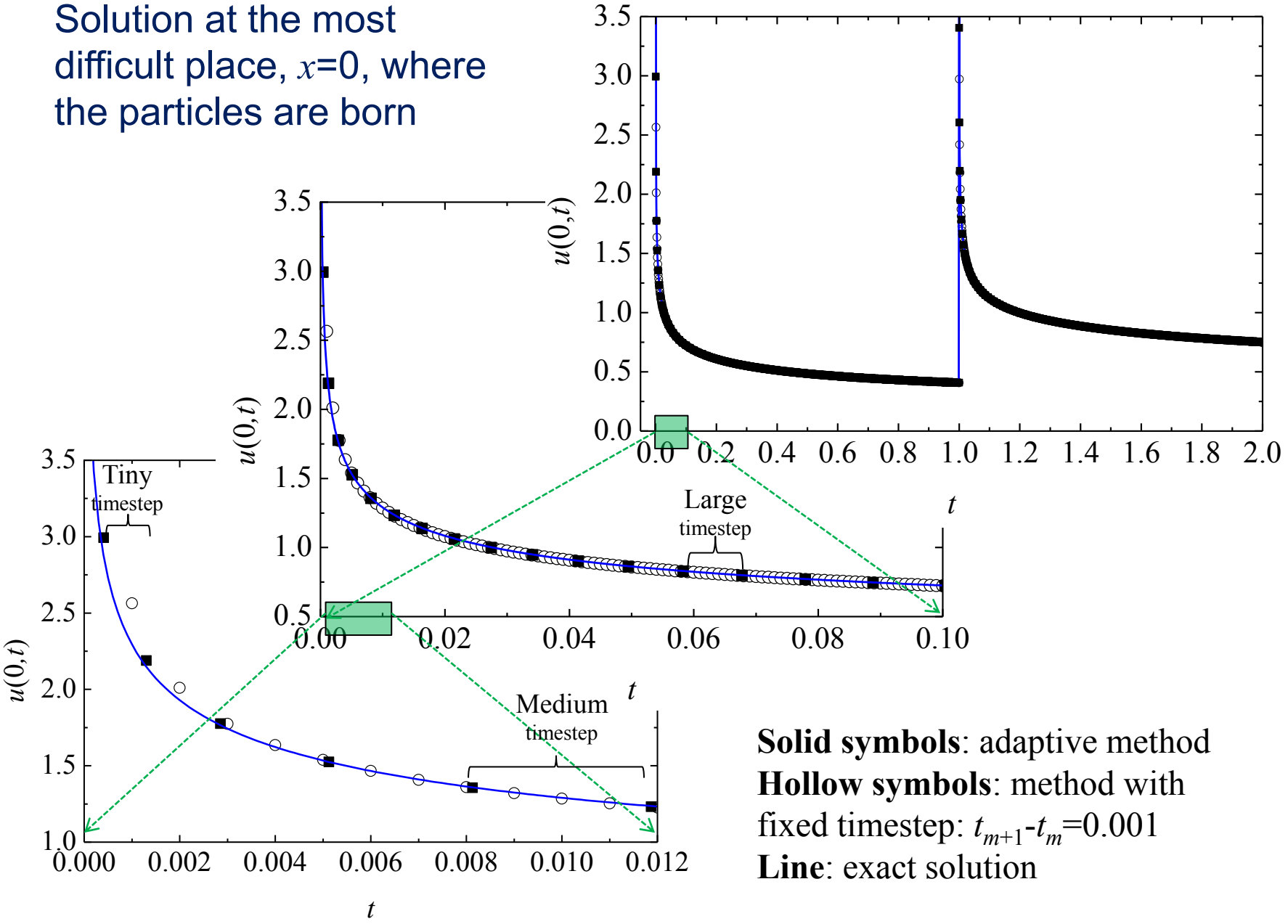

t=1-
65 timesteps
u(x,t)
x


t=1+
65 timesteps
u(x,t)
x


t=1.0004
66 timesteps
u(x,t)
x

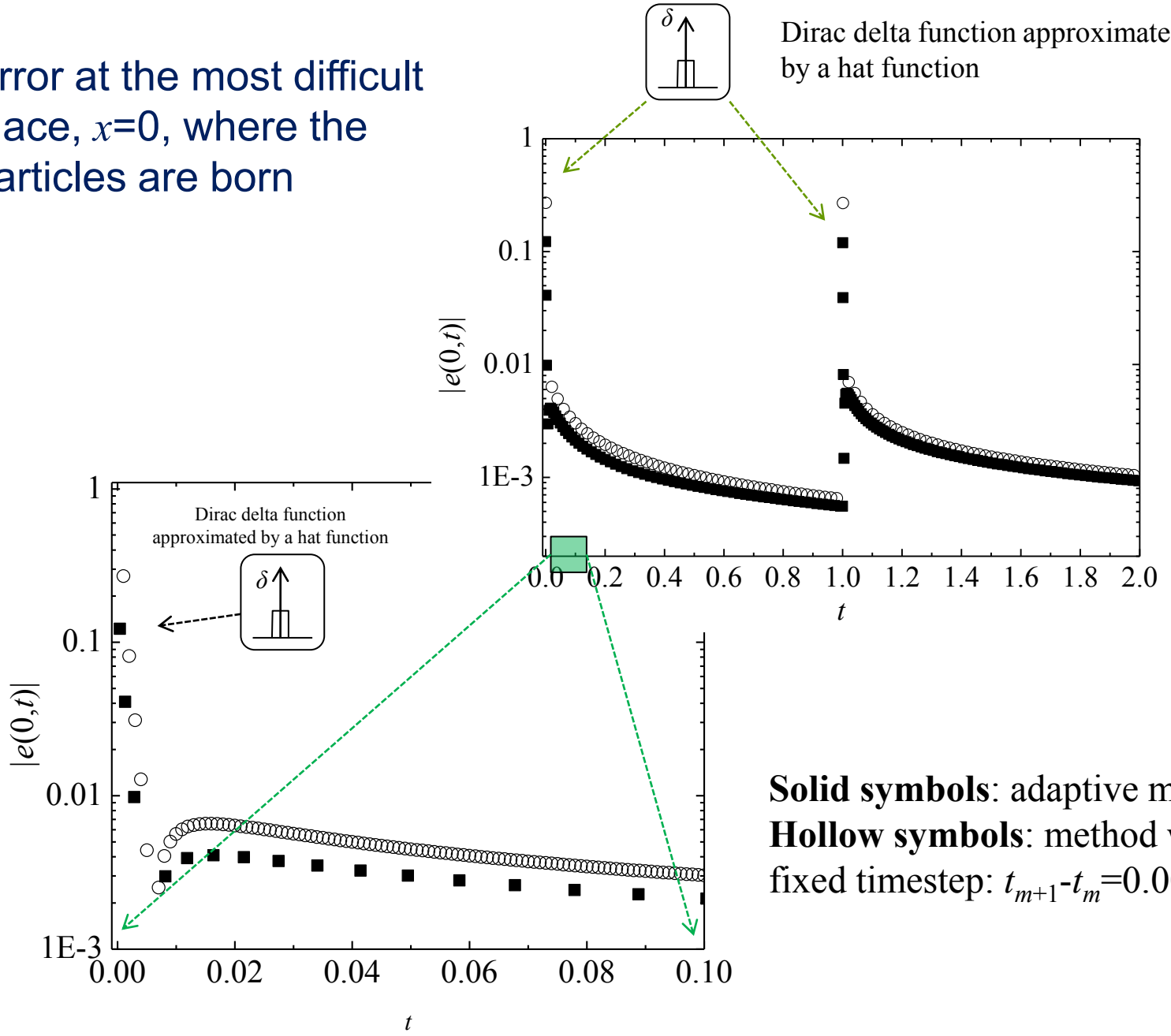

t=2-
141 timesteps
u(x,t)
x

Solution at the most difficult place, x=0, where the particles are born



Solid symbols: adaptive method
Hollow symbols: method with fixed timestep: t_{m+1}-t_m=0.001
Line: exact solution

Error at the most difficult place, x=0, where the particles are born

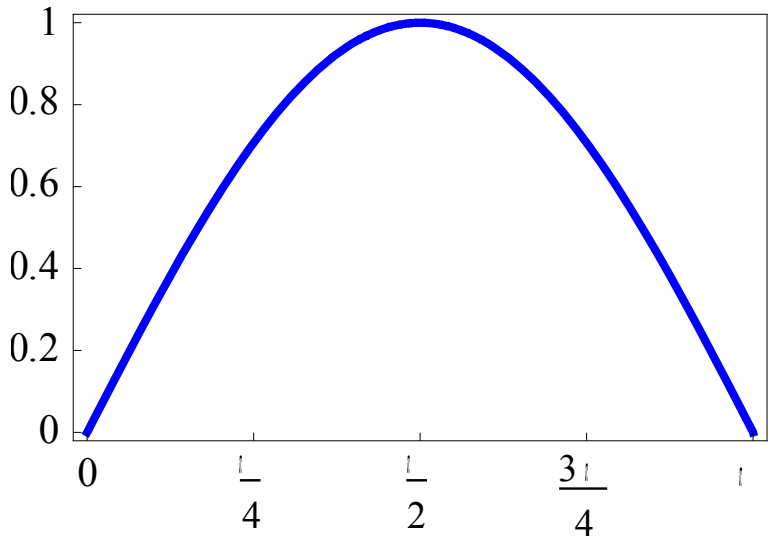


Solid symbols: adaptive method
Hollow symbols: method with fixed timestep: t_{m+1}-t_m=0.001

Another testbed problem. Another adaptive algorithm

A problem with initial condition easier to handle

$u(x,0)$



$u(x,t)$=?

$$\frac{\partial^\gamma u}{\partial t^\gamma} = K\frac{\partial^2 u}{\partial x^2}$$

$$0 \cdot x \cdot \pi$$

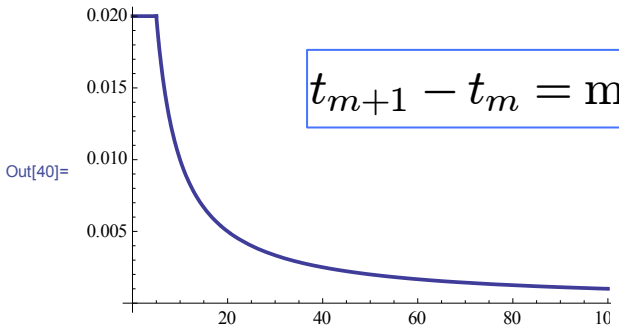
$$u(x=0,t) = u(x=\pi,t) = 0$$

$$u(x,0) = \sin x$$

$$K = 1$$

$$\Rightarrow \quad u(x,t) = E_\gamma(-t^\gamma)\sin(x)$$

How to choose the size of the timesteps?
Another adaptive algorithm



$$t_{m+1} - t_m = \min\left[10^{-4}\coth\left[|g(t_m)|/1000\right], 0.02\right]$$

Previous criterium: taylored for the problem at hand

Goal: a more general criterium.

A good ODE integrator should exert some adaptive control over its own progress, making frequent changes in its stepsize. **Usually the purpose of this adaptive stepsize control is to achieve some predetermined accuracy in the solution with minimum computational effort.** Many small steps should tiptoe through treacherous terrain, while a few great strides should speed through smooth uninteresting countryside.

Prest et al. *Numerical recipes*.

We explore an adaptive algorithm of classic style →

Another adaptive algorithm

We explore a straightforward algorithm of *classic* style

NUMERICAL RECIPES IN FORTRAN 77: THE ART OF SCIENTIFIC COMPUTING Section 16.2:

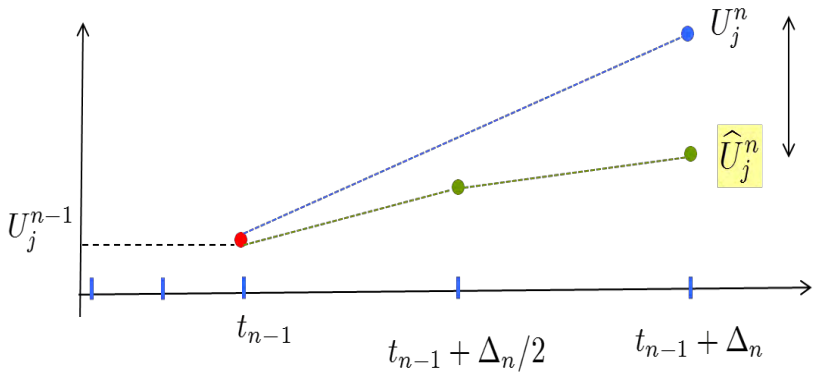
Usually the purpose of this adaptive stepsize control is to achieve some predetermined accuracy in the solution with minimum computational effort. Many small steps should tiptoe through treacherous terrain, while a few great strides should speed through smooth uninteresting countryside. […]

Implementation of adaptive stepsize control requires that the stepping algorithm return information about its performance, most important, **an estimate of its truncation error**. […] Obviously, the calculation of this information will add to the computational overhead, but the investment will generally be repaid handsomely

With fourth-order Runge-Kutta, …

… the most straightforward technique by far is **step doubling**. **We take each step twice, once as a full step, then, independently, as two half steps.** The difference between the two numerical estimates is a convenient indicator of truncation error. It is this difference that we shall endeavor to keep to a desired degree of accuracy, **neither too large nor too small**. **We do this by adjusting [*the timestep*]**

How to choose the size of the timesteps? A new algorithm



Step 2a → met. **accurate**

Step 2b→ met. **fast** and still accurate

1. Boostrap of step *n*: $\Delta_n = \Delta_{n-1}$ and $\left|\widehat{U}_k^{(n)} - U_k^{(n)}\right| \overset{?}{>} \text{tolerance} \equiv \tau$

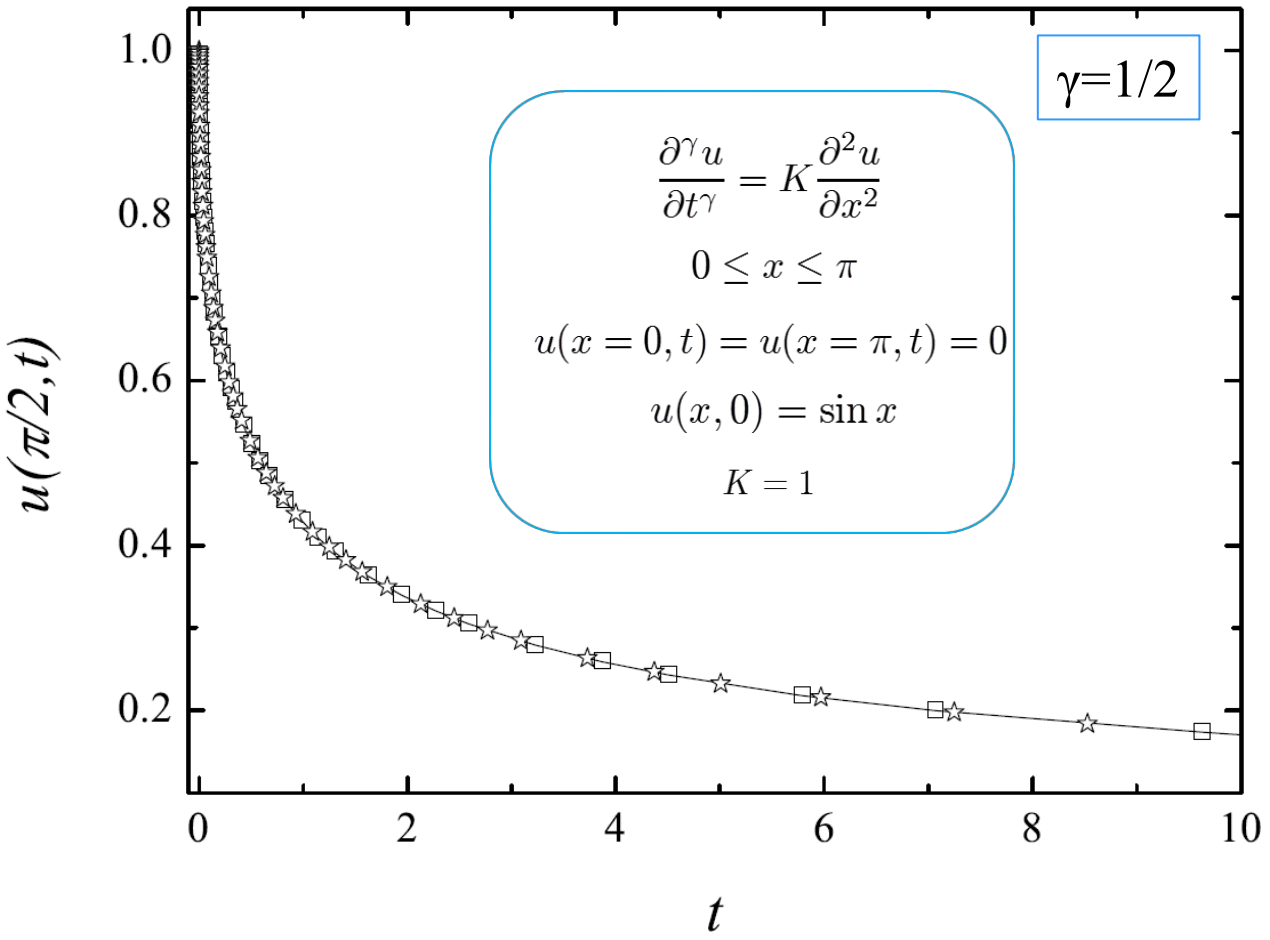
2a. True: then $\Delta_n \to \Delta_n/R$ until $\left|\widehat{U}_k^{(n)} - U_k^{(n)}\right| < \tau$ then $t_n = t_{n-1} + \Delta_n$

2b. False: then $\Delta_n \to R\Delta_n$ until $\left|\widehat{U}_k^{(n)} - U_k^{(n)}\right| > \tau$ then $t_n = t_{n-1} + \Delta_n/R$

3. Repeat [i.e., $n \to n+1$ and go to 1]

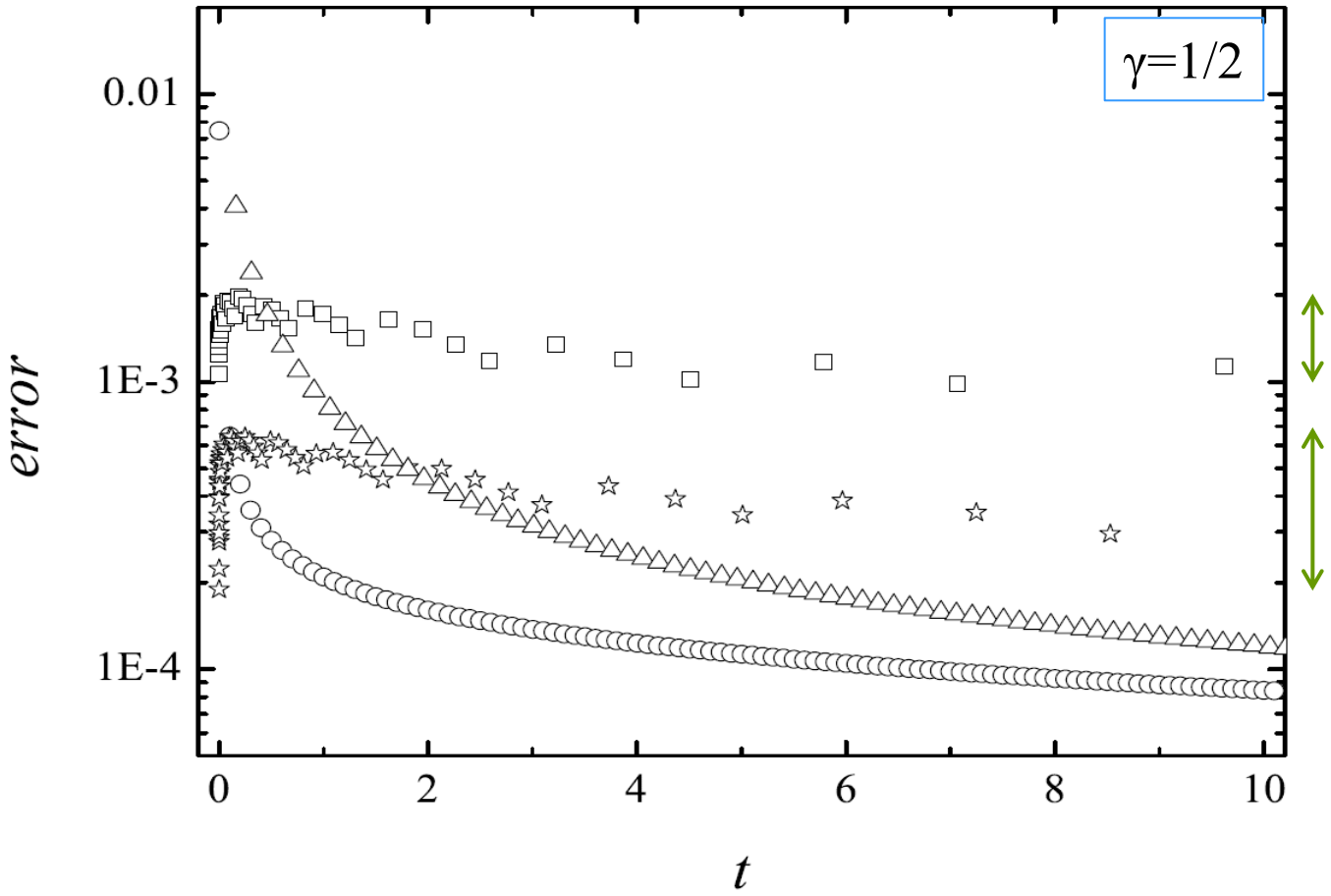
R: *brave* exploration coefficient→ we use *R*=2

Numerical results



Solid line: exact solution; squares: adaptive numerical solution up to $n = 45$ ($t_{45} = 9.6268$) and tolerance $\tau = 5 \times 10^{-4}$; stars: numerical solution up to $n = 112$ ($t_{112} = 9.1690$) and $\tau = 10^{-4}$. In both cases $\Delta_0 = 0.01$ and $\Delta x = \pi/40$.

Numerical results



Errors in the adaptive algorithm are even (great!, compare with those for fixed timesteps)

Numerical errors at the mid-point of (i) the adaptive method with $\tau = 5\times10^{-4}$, $\Delta_0 = 0.01$ (squares, 45 timesteps, CPU time ≈ 0.15s); (ii) the adaptive method with $\tau = 10^{-4}$ and $\Delta_0 = 0.01$ (stars, 112 timesteps, CPU time ≈ 0.75s); (iii) the method with constant timesteps of size $\Delta_n = 0.01$ (triangles, 1010 timesteps, CPU time ≈ 440s); and (iv) the method with uniform timesteps of size $\Delta_n = 0.001$ (circles, 10100 timesteps, CPU time ≈ 43800s). In all cases $\gamma = 1/2$ and $\Delta x = \pi/40$.

Does the method always work?

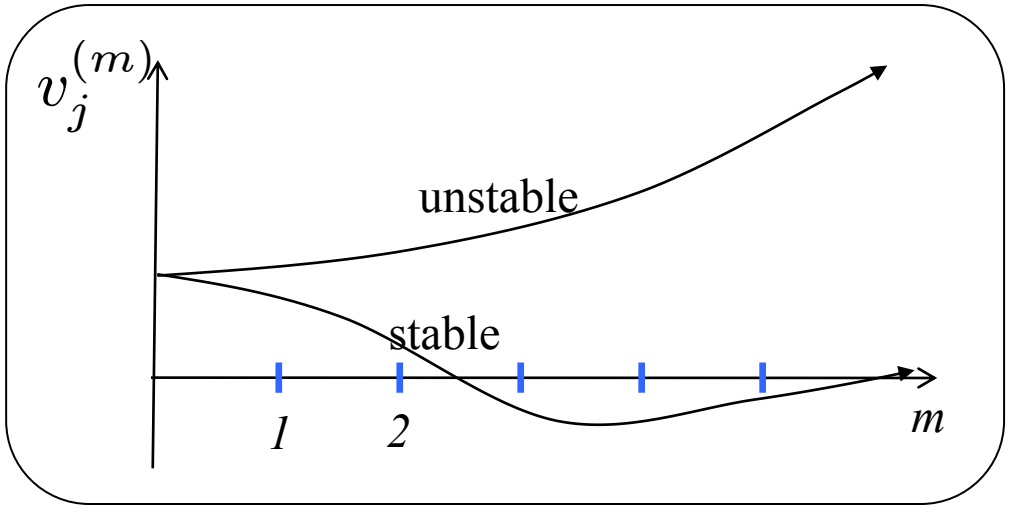
Stability

Do the inevitable numerical perturbations (numerical noise, round-off errors) ruin the method? → **How do the perturbations evolve?**

$$\widehat{U}_j^{(m)} - U_j^{(m)} = v_j^{(m)}$$

$\widehat{U}_j^{(m)}$: perturbed solution

$v_j^{(m)}$: perturbation



Linear

$$AU^{(n)} = \mathcal{M}U^{(n)} + \tilde{F}^{(n)} \quad \Longrightarrow \quad Av^{(n)} = \mathcal{M}v^{(n)}$$

Eq. for the evolution of the perturbation

Does the method always work?
Stability

Von Neumann stability analysis: **(Yuste-Acedo, arXiv:cs/0311011,** *SIAM J. Num. Anal.* 05 **)**
Suscessful for numerical methods for fractional equations with Riemman-Liouville and Pareto derivatives and for Grünwald-Letnikov, L1 and L2 discretization schemes, and many others

Von Neumann procedure:

1: $$v_j^{(n)} = \sum_q \xi_q^{(n)} e^{iqj\Delta x}$$

2: Stability analysis of ***a generic subdiffusive mode***: $\xi_q^{(n)} e^{iqj\Delta x}$

$$Av^{(n)} = \mathcal{M}v^{(n)}$$

$$A\left[\xi_q^{(n)} e^{iqj\Delta x}\right] = \mathcal{M}\left[\xi_q^{(n)} e^{iqj\Delta x}\right]$$

Equation for the evolution of $\xi_q^{(n)}$: $$\mathcal{F}\left[\xi_q^{\{n\}}, S(q)\right] = 0$$

Does the method always work? **Stability**

First example: von-Neumann stability analysis of the Yuste-Acedo method (*SIAM J. Num. Anal.* 05 , fixed timesteps) for Riemann-Liouville FDE

RL derivative

$$S(q) \equiv 4\frac{(\Delta t)^\gamma}{(\Delta x)^2}\sin^2\left(\frac{q\Delta x}{2}\right)$$

Grunwald-Letnikov coefficients

$$\frac{\partial}{\partial t}u(x,t) = K_\gamma\ {}_0D_t^{1-\gamma}\frac{\partial^2}{\partial x^2}u(x,t) \Longrightarrow \xi^{(n+1)} = \xi^{(n)} - S(q)\sum_{k=0}^{n}\omega_k^{(1-\gamma)}\xi^{(n-k)} \quad (*1)$$

$$\Updownarrow$$

$$\mathcal{F}\left[\xi^{\{n+1\}}, S(q)\right] = 0$$

- Eq. of ***evolution*** of the ***amplitude*** of a generic mode.
- Similar to those of (non-fractional) multistep (multilevel) schemes.
- This equation and/or fractional difference methods could be seen as *particular cases of multilevel schemes where the number of levels* increases *with time.*

Does the method always work? Stability

First example: Yuste-Acedo method (*SIAM J. Num. Anal. 05* , fixed timesteps) for RL FDE

$$(*1) \qquad \xi^{(m+1)} = \xi^{(m)} - S(q)\sum_{k=0}^{m} \omega_k^{(1-\gamma)} \xi^{(m-k)} \quad \equiv \quad \mathcal{F}\left[\xi^{\{n\}}, S(q)\right] = 0$$

(*1) Stable if the n roots of

Stability polynomial $z^n = z^{n-1} - S(q)\sum_{k=0}^{n-1} \omega_k^{(1-\gamma)} z^{n-1-k}$ satisfy $|z_n| \leq 1$

$$S(q) = S(q,n) \equiv \frac{-z^n + z^{n-1}}{\sum_{k=0}^{n-1} \omega_k^{(1-\gamma)} z^{n-1-k}}$$

Stability region: region (in the complex plane) formed by those values of $S(q,n)$ for which $|z_n| \leq 1$

$$S(q) \equiv 4\frac{(\Delta t)^\gamma}{(\Delta x)^2}\sin^2\left(\frac{q\Delta x}{2}\right)$$

Boundary of the stability region:

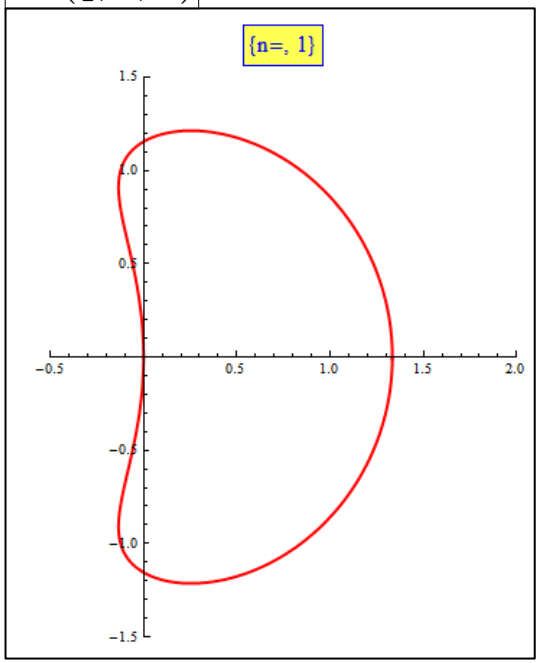
$$\begin{cases} z = e^{i\theta} \\ S(q) \to S_B(q,\theta) \quad \text{with} \quad \theta = 0 \to \theta = 2\pi \end{cases}$$

Does the method always work? Stability

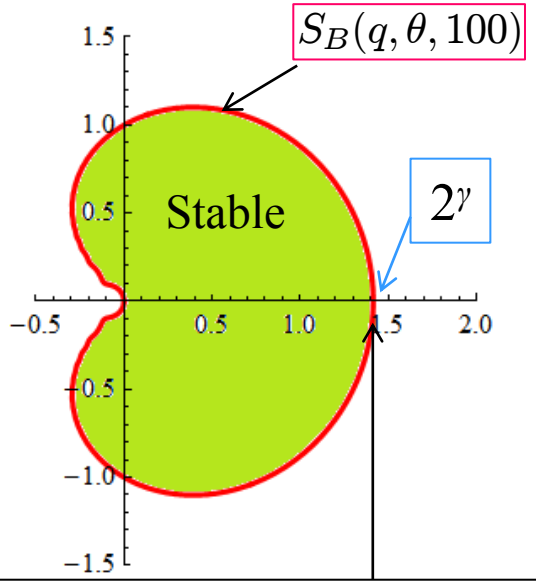
First example: Yuste-Acedo method (*SIAM J. Num. Anal.* 05 , fixed timesteps, explicit) for RL FDE

$$S(q,n) \equiv \frac{-z^{n+1}+z^n}{\sum_{k=0}^{n}\omega_k^{(1-\gamma)} z^{n-k}} \xrightarrow{z=e^{i\theta}} S_B(q,\theta,n) \equiv \frac{-e^{i(n+1)\theta}+e^{in\theta}}{\sum_{k=0}^{n}\omega_k^{(1-\gamma)} e^{i(n-k)\theta}}$$

$S_B(q,\theta,n)$



γ=1/2



$$S(q) \equiv 4\frac{(\Delta t)^\gamma}{(\Delta x)^2}\sin^2\left(\frac{q\Delta x}{2}\right)$$

Numerical method for RL-FDE: stable when

$$4\frac{(\Delta t)^\gamma}{(\Delta x)^2} \cdot \quad S_\times = 2^\gamma$$

Yuste-Acedo arXiv:cs/0311011v1 [cs.NA] (2003)

Does the method always work? Stability

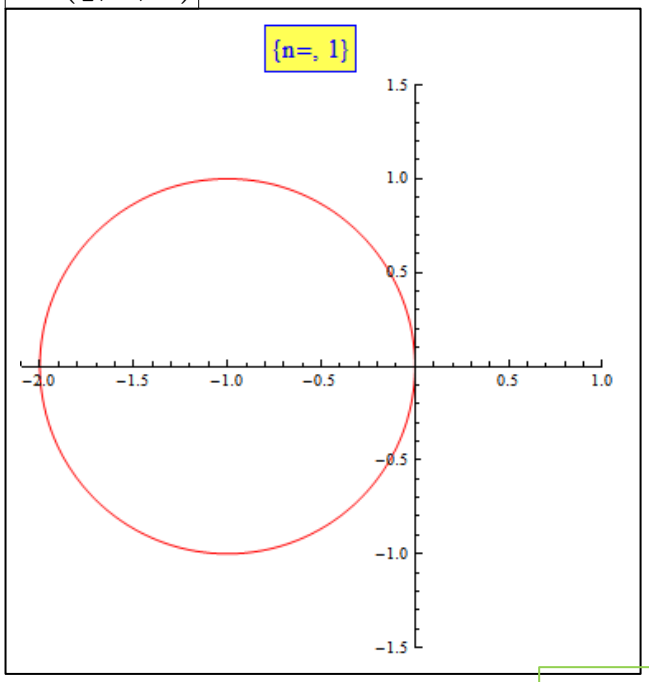
Second example: von-Neumann stability analysis of the Liu-Zhuang-Anh-Turner method* (ANZIAM J. 47 (2006) ,fixed timesteps, implicit) for Caputo FDE

*=present method for fixed timesteps

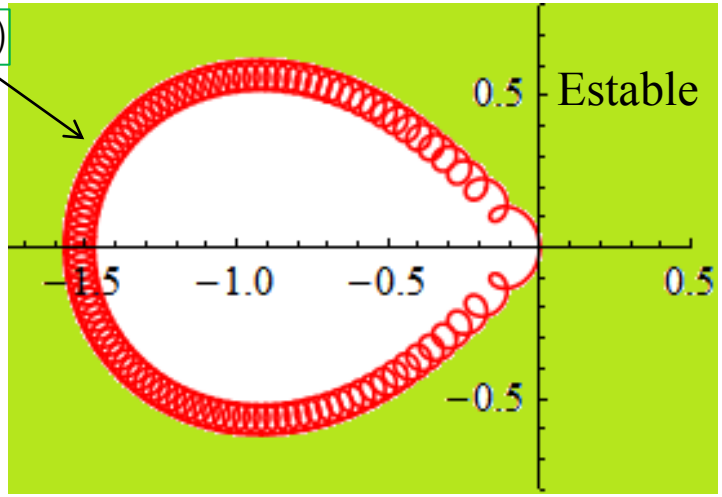
$$\frac{\partial^\gamma u}{\partial t^\gamma} = K\frac{\partial^2 u}{\partial x^2} \quad \Longrightarrow \quad S(q,n) \equiv \frac{-z^n + z^{n-1}}{\sum_{k=1}^{n-1}\left[(k+1)^{1-\gamma} - k^{1-\gamma}\right] z^{n-k}}$$

Stability polynomial

$S_B(q,\theta,n)$



$S_B(q,\theta,100)$



Numerical method stable for all S(q)>0
→ unconditionally stable

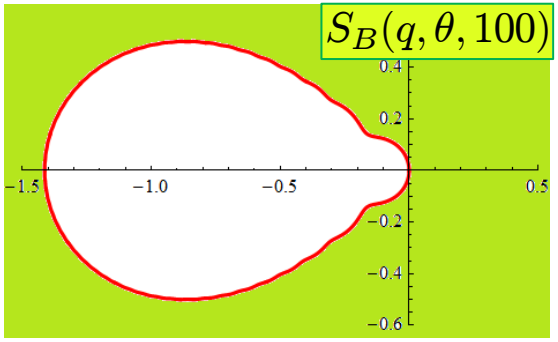
$$S(q) \equiv 4\frac{(\Delta t)^\gamma}{(\Delta x)^2}\sin^2\left(\frac{q\Delta x}{2}\right)$$

Does the method always work? **Stability**

A zoo of stability regions

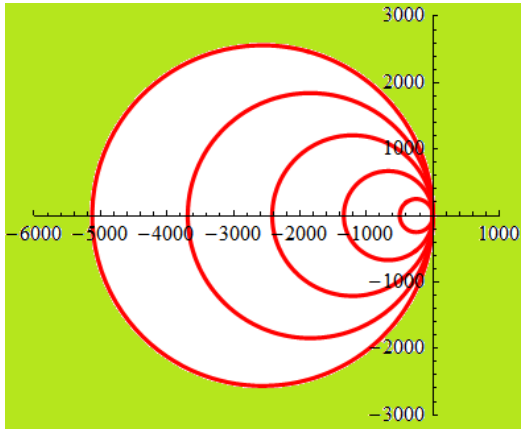
RL FDE, weighted-average method,
(Yuste, Journal of Computational Physics (2006))

Fully implicit



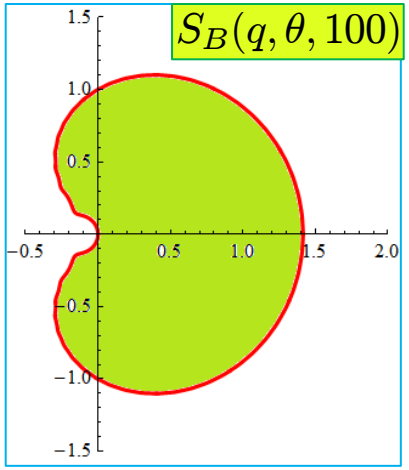
Unconditionally stable

(fractional) Crank-Nicholson



Unconditionally stable

Explicit

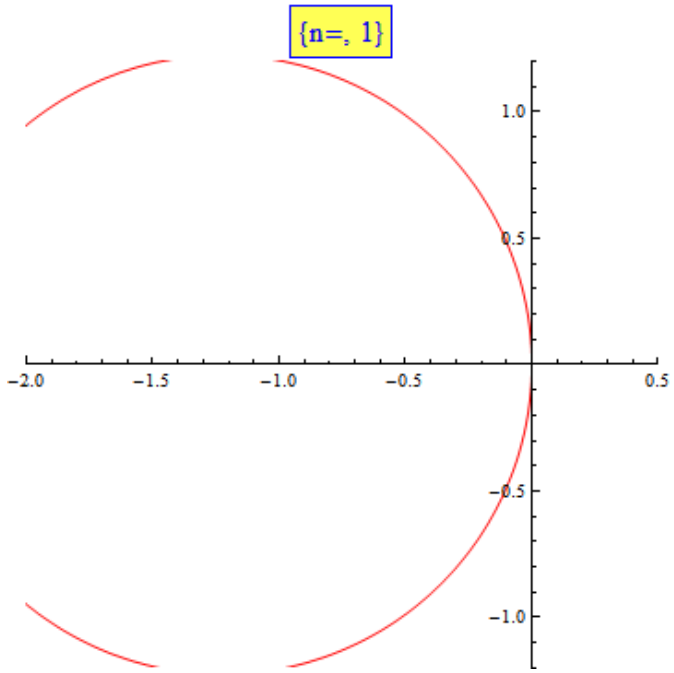


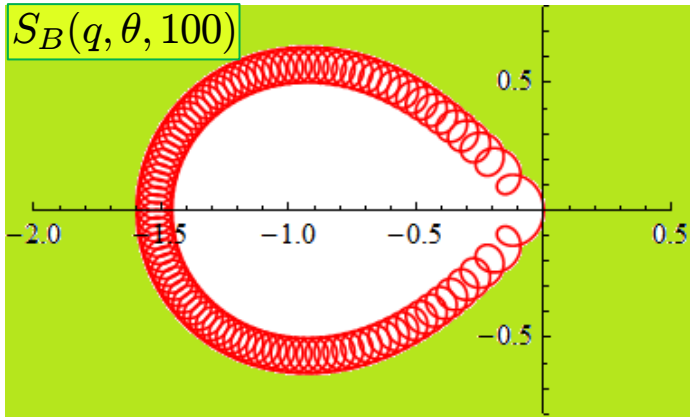
conditionally stable

Does the method always work? **Stability**

Does the (fractional) von-Neumann stability analysis work for *variable* timesteps?

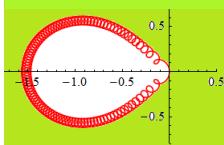
Present method (Caputo FDE, implicit) with *variable* timesteps: $\Delta t_m = (m+2)/10$





Unconditionally stable

Fixed timesteps



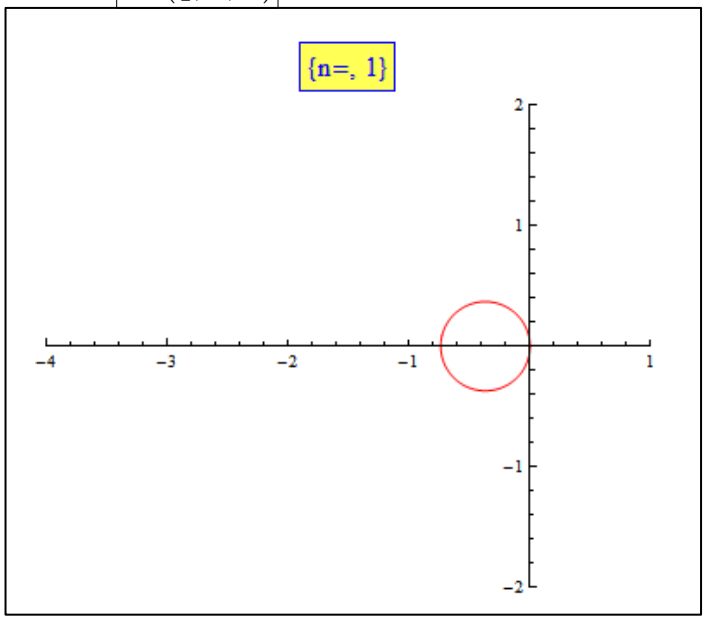
Does the method always work? **Stability**

Does the (fractional) von-Neumann stability analysis work for *variable* timesteps?

Present method (Caputo FDE, implicit) with random *variable* timesteps: $\Delta t_m = r/10$

r=random number, uniform in [0,2]

$S_B(q,\theta,n)$

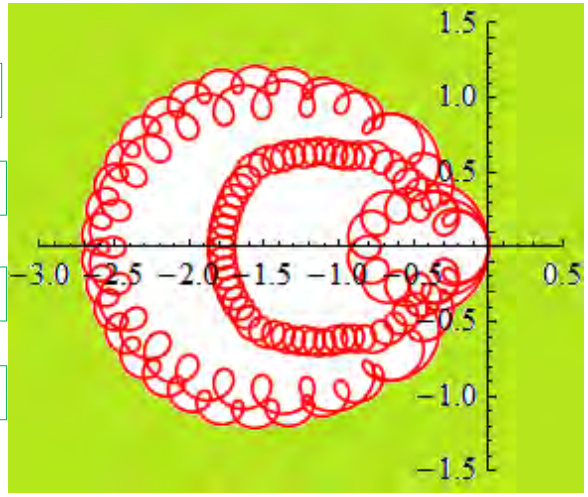


$S_B(q,\theta,10)$

$S_B(q,\theta,20)$

$S_B(q,\theta,30)$

$S_B(q,\theta,50)$



Unconditionally stable

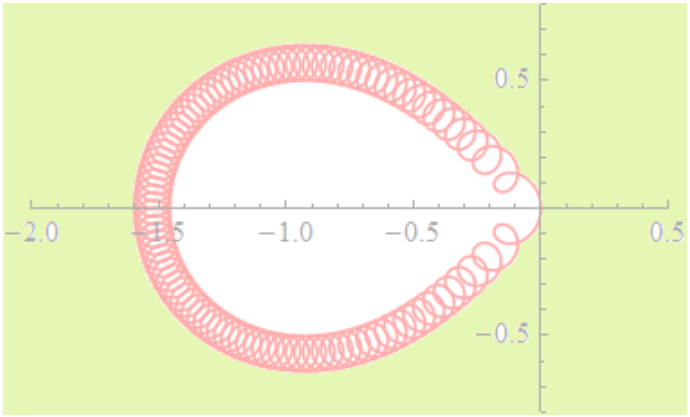
Does the method always work?
Stability

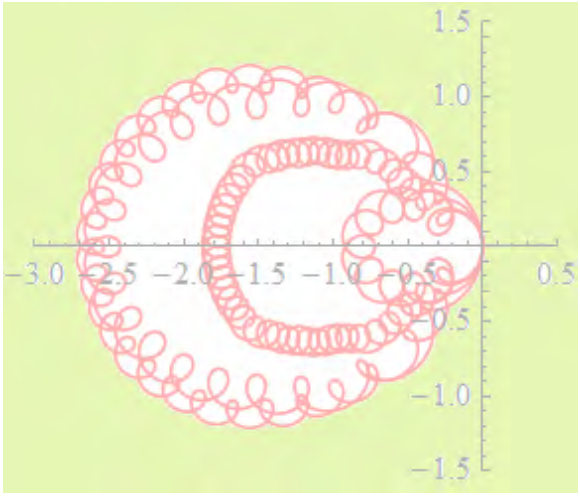
The present implicit method for the FDE in the Caputo form with variable timesteps is

***unconditionally* stable for *any* choice of timesteps**

(**Yuste&Quintana-Murillo**, Computer Physics Communications, Vol. 183, December 2012)

It is proved there that $\left\| v^{(n)} \right\|_2 \cdot \left\| v^{(0)} \right\|_2$ always!





Adaptive finite difference method for Diffusion-wave equation

$$\frac{\partial^\gamma u}{\partial t^\gamma} = K \frac{\partial^2 u}{\partial x^2} + F(x,t)$$

$$1 < \gamma < 2$$

Finite difference method: Discretization of the FPDE

diffusion-wave

$$\partial \equiv \frac{\partial^\gamma}{\partial t^\gamma} - K\frac{\partial^2}{\partial x^2} \longrightarrow \delta \equiv \delta_t^\gamma - K\delta_x^2$$

$$\frac{\partial^2 u}{\partial x^2} \longrightarrow \delta_x^2 u(x_j,t) = \frac{u(x_{j+1},t) - 2u(x_j,t) + u(x_{j-1},t)}{(\Delta x)^2}$$

Discretization of the Laplacian: three point centered formula

$1<\gamma<2$: diffusion-wave equation

$$\frac{\partial^\gamma u}{\partial t^\gamma} \longrightarrow \delta_t^\gamma u(x,t_n) = \frac{1}{\Gamma(3-\gamma)}\sum_{m=0}^{n-1}\left\{A_{m,n}^{(\gamma)}\left[u(x,t_{m+1}) - u(x,t_m)\right] - B_{m,n}^{(\gamma)}\left[u(x,t_m) - u(x,t_{m-1})\right]\right\}$$

L2 discretization of the fractional Caputo derivative

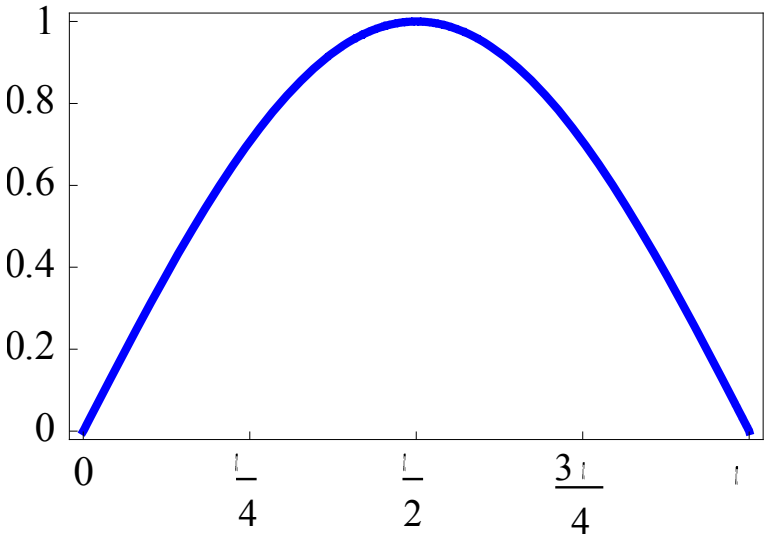
$$-S_n U_{j+1}^n + (1+2S_n)U_j^n - S_n U_{j-1}^n = \mathcal{M}U_j^n + \tilde{F}(x_j,t_n)$$

$$\mathcal{M}U_j^n \equiv U_j^{(n-1)} + \frac{t_n - t_{n-1}}{t_{n-1} - t_{n-2}}\left[U_j^{(n-1)} - U_j^{(n-2)}\right] - \sum_{m=0}^{n-2}\left(\hat{A}_{m,n}\left[U_j^{(m+1)} - U_j^{(m)}\right] - \hat{B}_{m,n}\left[U_j^{(m)} - U_j^{(m-1)}\right]\right)$$

$$S_n = \frac{\Gamma(3-\gamma)K(t_n - t_{n-2})}{2(t_n - t_{n-1})^{1-\gamma}(\Delta x)^2}$$

The testbed problem

$u(x,0)$



$u(x,t)$=?

$$\frac{\partial^\gamma u}{\partial t^\gamma} = K\frac{\partial^2 u}{\partial x^2}$$

$$0 \cdot \quad x \cdot \quad \pi$$

$$u(x=0,t) = u(x=\pi,t) = 0$$

$$u(x,0) = \sin x$$

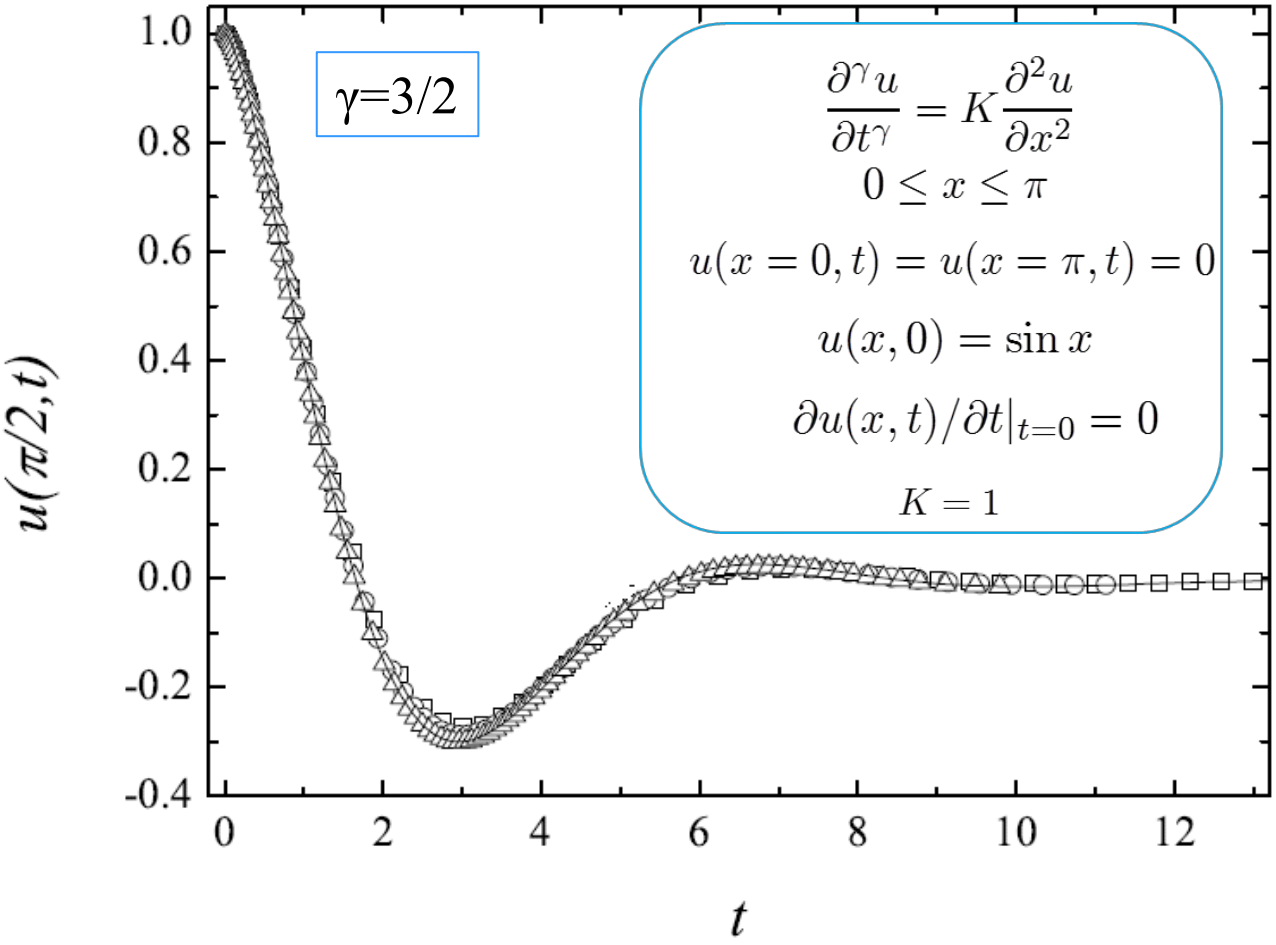
$$\partial u(x,t)/\partial t|_{t=0} = 0$$

$$K = 1$$

$$\Rightarrow \quad u(x,t) = E_\gamma(-t^\gamma)\sin(x)$$

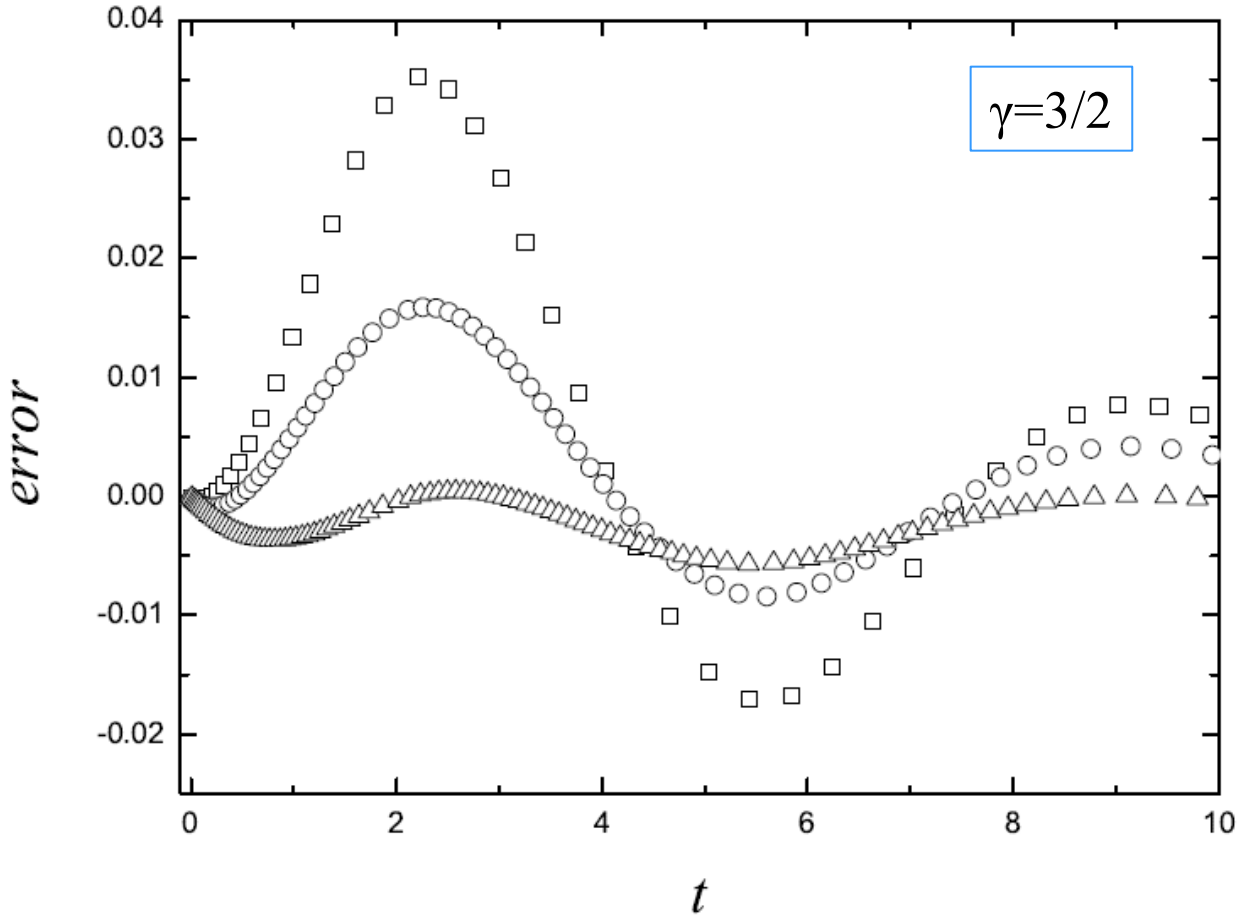
Step doubling algorithm

Numerical results



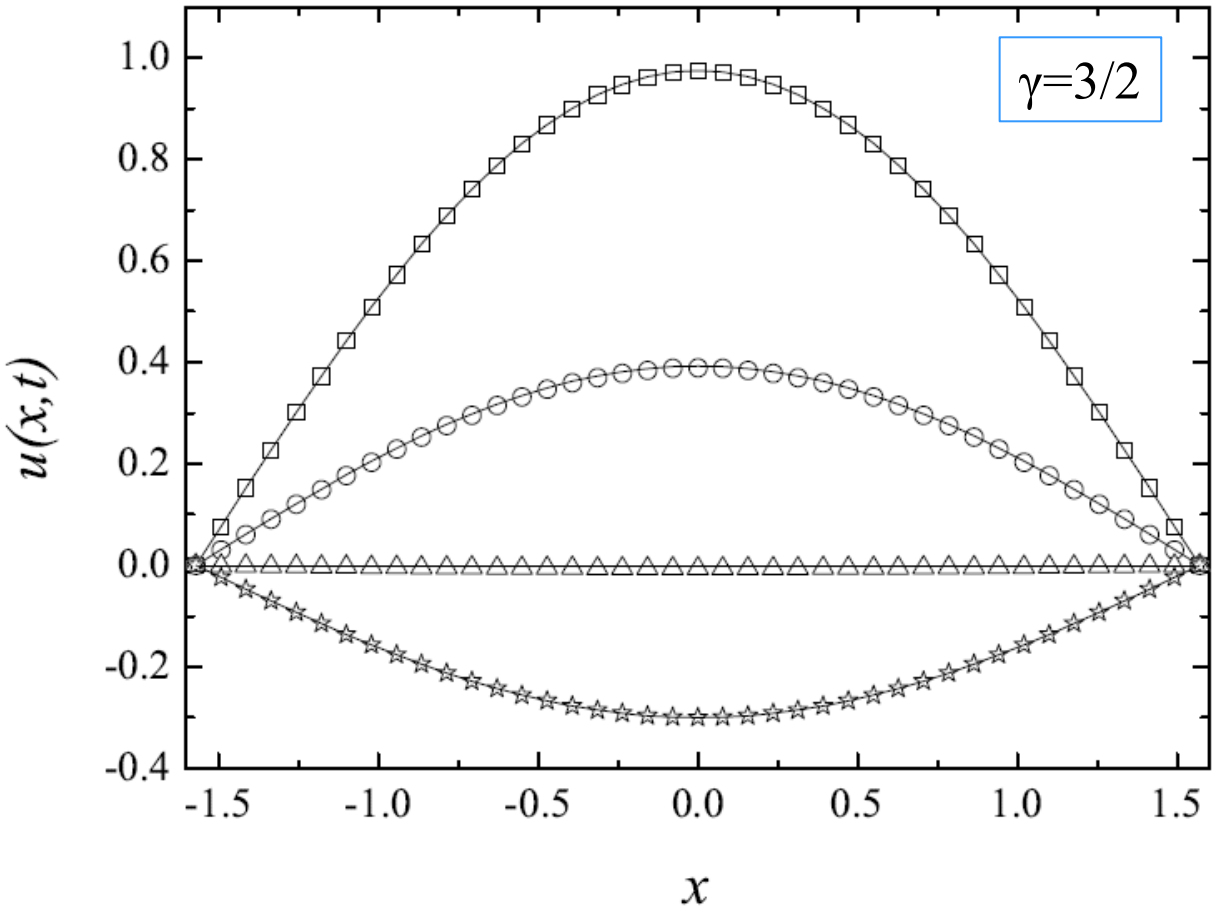
Solid line: exact solution; symbols: adaptive method with $\tau = 10^{-4}$ up to time $t_{100} = 13.1958$ (squares), $\tau = 10^{-5}$ up to time $t_{150} = 11.3309$ (circles), and $\tau = 10^{-6}$ up to time $t_{300} = 9.9783$ (triangles). In all cases $\Delta x = \pi/40$ and $\Delta_0 = 0.01$

Numerical results



Numerical errors at the mid-point, $|u(\pi/2, t_n) - U_k^n|$, of the adaptive method for $\tau = 5 \times 10^{-4}$ (squares), $\tau = 10^{-5}$ (circles), and $\tau = 10^{-6}$ (triangles). In all cases $\Delta_0 = 0.01$ and $\Delta x = \pi/40$.

Numerical results



Exact solution (lines) and adaptive numerical solution (symbols) of the diffusion-wave problem described in the main text for $\gamma = 3/2$ and $t_{16} = 0.105$ (squares), $t_{84} = 1.006$ (circles), $t_{156} = 3.015$ (stars), and $t_{233} = 5.741$ (triangles), with $\tau = 10^{-6}$, $\Delta_0 = 0.01$ and $\Delta x = \pi/40$.

Numerical results. Diffusion-wave equation
Some remarks

Disappointing results (in comparison with those for subdiffusion equations)

Results **worsens** when the lengths of two successive timesteps, Δ_n and Δ_{n-1}, are too different

We cap the ratio between the lengths of two successive timesteps, [$|\Delta_n/\Delta_{m-1}|$ or $|\Delta_{n-1}/\Delta_n|$] to be smaller than 1.1.

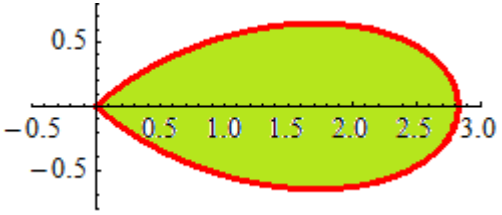
R=1.02 (really timid exploration coefficient)

Worsens?→ Is the method stable?

Numerical tests

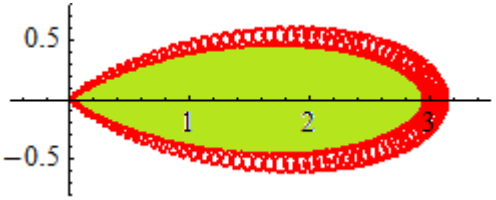
We have carried out extensive calculations and considered a large variety of timestep functions (including random distributions), and we have always found well-behaved (stable) numerical solutions

Diffusion-wave equation. Boundaries of stability for some methods with *fixed* timesteps



Explicit

Gorenflo,Mainardi, Moretti & Paradisi 2002, Nonlinear Dyn.; also, Quintana-Murillo & Yuste, 2009, Physica Scripta



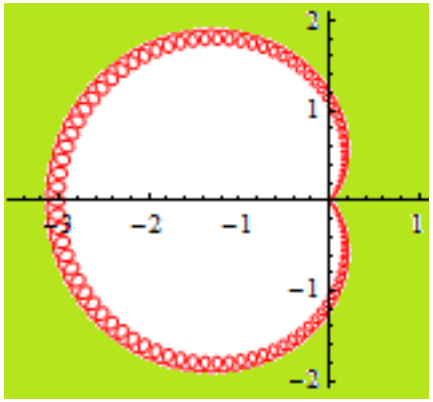
Explicit

Quintana-Murillo & Yuste.,2011. J. Comp.Nonlin. Dyn.

⇩

The stability analysis works fine…

… but →

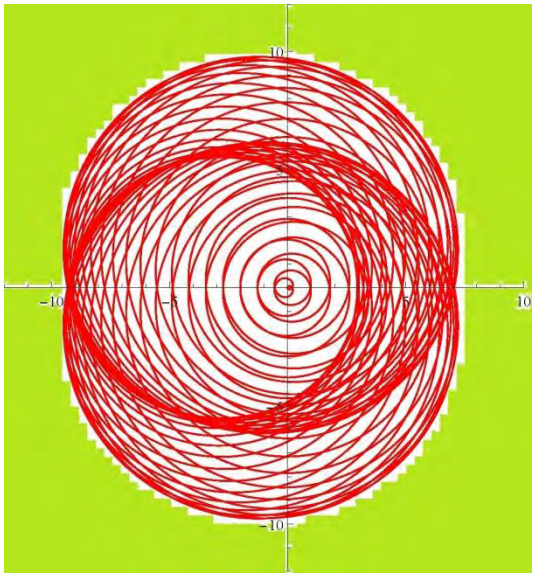


Implicit

Present method for fixed timesteps

Diffusion-wave equation. Boundaries of stability for *variable* timesteps

$S_B(q,\theta,n)$ n=50



An example

$$\Delta t_m = (m+2)/100$$

?

However→

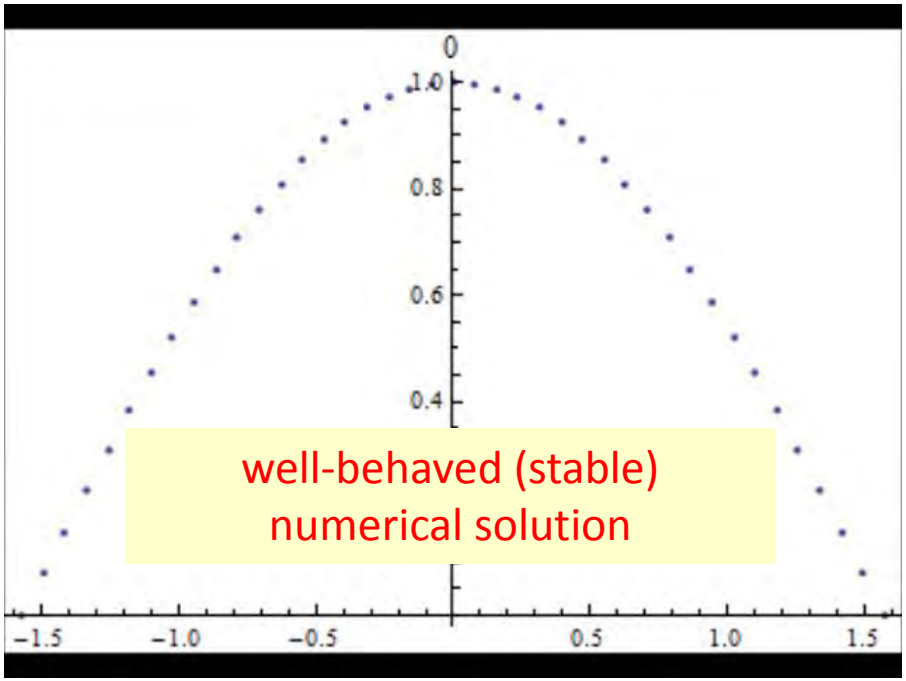
$$\frac{\partial^\gamma u}{\partial t^\gamma} = K\frac{\partial^2 u}{\partial x^2}$$

$$0 \le x \le \pi$$

$$u(x=0,t) = u(x=\pi,t) = 0$$

$$u(x,0) = \sin x$$

$$K = 1$$



well-behaved (stable) numerical solution

$$\Delta t_m = (m+2)/100$$

$$S(q) \equiv 4\frac{(\Delta t)^\gamma}{(\Delta x)^2}\sin^2\left(\frac{q\Delta x}{2}\right)$$

Remarks and conclusions

- Computational cost of fractional difference methods → huge
- Different time scales → usual

⇩

- *Adaptive* finite difference method with non-uniform timesteps → {very convenient≈a must}
- von Neumann stability analysis for homogeneous timesteps → a breeze
- von Neumann stability analysis for variable timesteps:
  - for subdifusion equations → works smoothly
  - for diffusion-wave equations → **?**
- Adaptive "step doubling method" → fast and accurate (for subdiffusion equations)