Basics of surface wave simulation

L. Ridgway Scott

Departments of Computer Science and Mathematics, Computation Institute, and Institute for Biophysical Dynamics,

University of Chicago

People in science

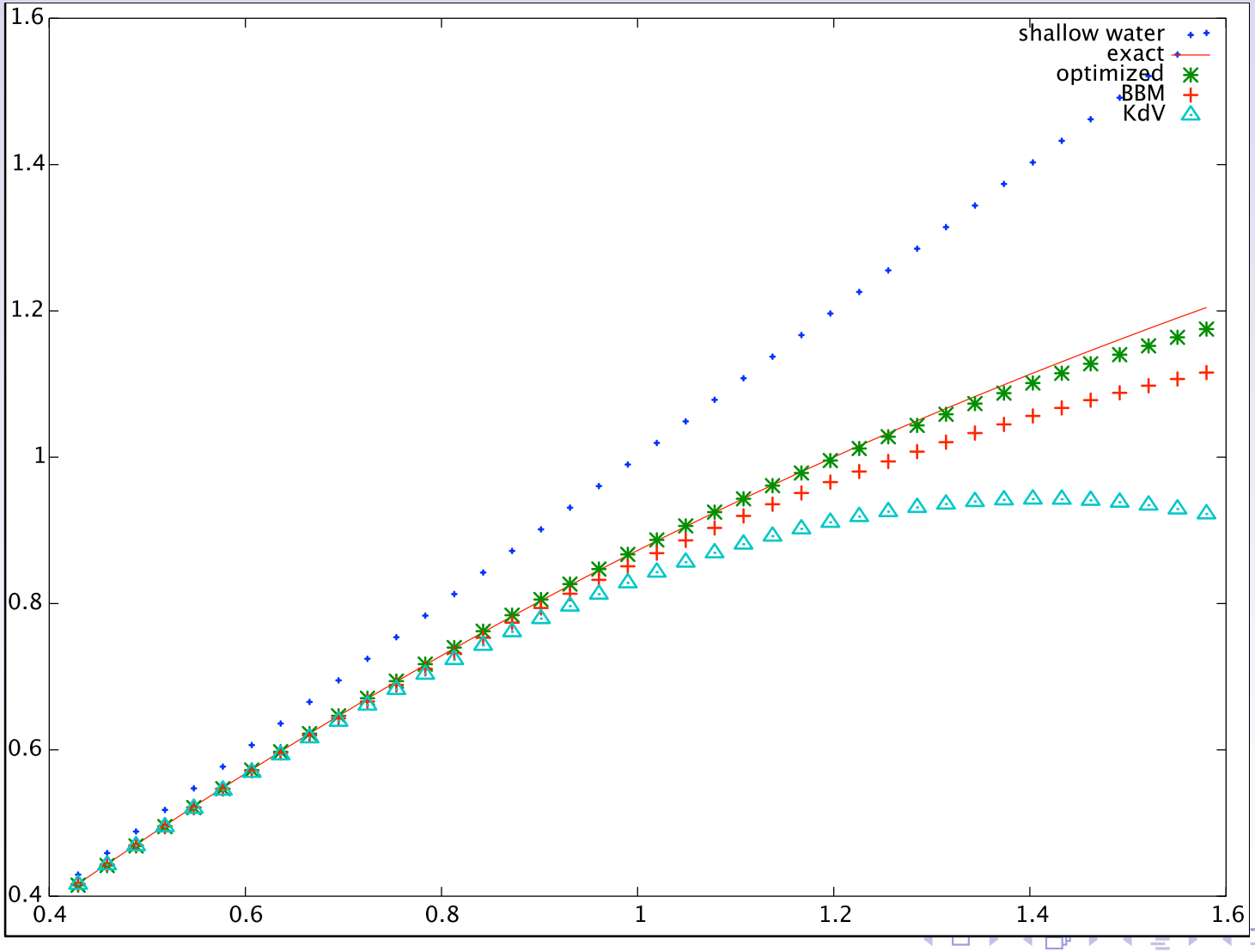
Fritz Joseph Ursell FRS (28 April 1923 — 11 May 2012)

Sir George Gabriel Stokes, 1st Baronet FRS (13 August 1819 — 1 February 1903), was senior wrangler at Cambridge in 1841. Together with Seidel, he is credited with for identifying the concept of uniform convergence as key to convergence of sums of functions.

Niels Henrik Abel (1802—1829) wrote, in a letter to a colleague in 1824, “in analysis one is largely concerned with functions that can be represented by power-series. As soon as other functions enter—and this happens rarely—then [induction] does not work any more and an infinite number of incorrect theorems arise from false conclusions”

Dispersion relations

The dispersion relation for (linear) water waves is $\omega(k) = \sqrt{k \tanh(k)}$. This can be compared to the linearized model equations: $k = 2\pi/\lambda$ ($k \leq 0.4$ iff $\lambda \geq 15.7$).



Dispersion equations

The dispersion relation for water waves is $\omega(k) = \sqrt{k \tanh(k)}$.

The dispersion relation for shallow water is $\omega(k) = k$.

The dispersion relation for BBM is

$$\omega(k) = \frac{k}{1 + \frac{1}{6}k^2} \tag{1}$$

The dispersion relation for KdV is

$$\omega(k) = k\left(1 - \tfrac{1}{6}k^2\right) \tag{2}$$

The dispersion relation for the optimized model [BPS81] is

$$\omega(k) = \frac{0.9898k}{1 + 0.1325k^2} \tag{3}$$

Discretizing 2nd order derivative

Defining a matrix for solving $\alpha u - u'' = f$:

```
kc=0;
for k=1:nr;
kc=kc+1; iv(kc)=k; jv(kc)=k;
sv(kc)=alfa+(2/(dx*dx));
end
for k=2:nr;
kc=kc+1; iv(kc)=k-1; jv(kc)=k;
sv(kc)=-(1/(dx*dx));
end
for k=2:nr;
kc=kc+1; iv(kc)=k; jv(kc)=k-1;
sv(kc)=-(1/(dx*dx));
end
svm=sparse(iv,jv,sv);
```

BBM difference method

Using the sparse matrix fod associated with the first-order finite difference

and the sparse matrix svm associated with solving $\alpha u - u'' = f$, the time-stepping for BBM looks like

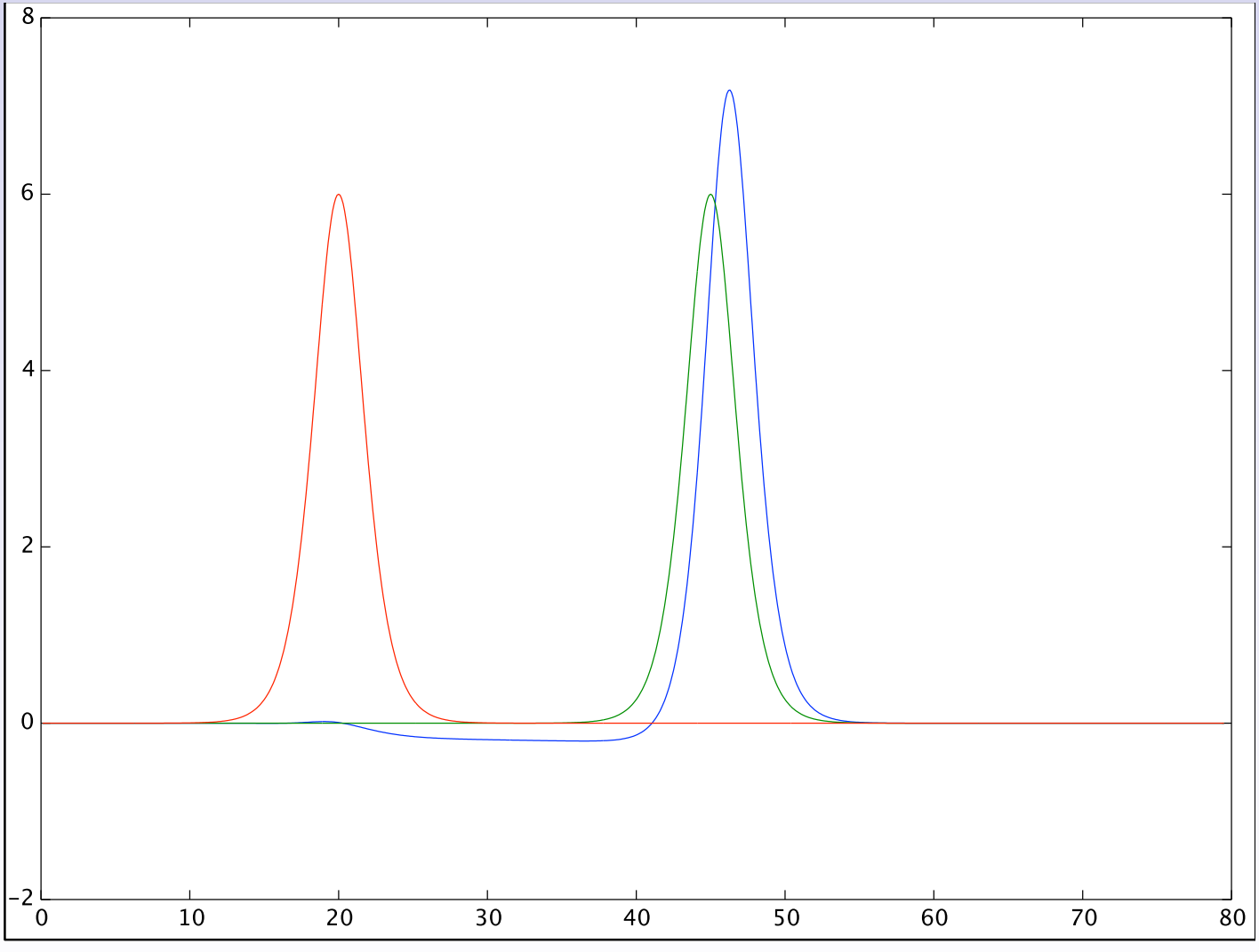
...
cfl=dt/dx
for k=1:nts
yu=yu-cfl*svm\ (fod(yu + yu .* yu));

end

We can test this scheme using a solitary wave.

Numerical error

There is substantial numerical error in a first-order time-stepping scheme (dx=.1, dt=.01)



First-order versus second-order

Define $\hat{y}_{i+1} = y_i + \Delta t\ f(y_i)$. Then

explicit Euler: $y_{i+1} = \hat{y}_{i+1}$

predictor corrector: $y_{i+1} = \frac{1}{2}(\hat{y}_{i+1} + y_i + \Delta t\ f(\hat{y}_{i+1}))$

	relative max error	dx	dt	cpu seconds
explicit	4.3646e-02	1.0e-01	1.0e-03	0.5
Euler	2.2438e-02	5.0e-02	5.0e-04	1.6
	4.4059e-03	2.5e-02	1.0e-04	13.0
predictor	1.9110e-03	1.0e-01	1.0e-02	0.2
corrector	4.9510e-04	5.0e-02	5.0e-03	0.5
	8.0939e-05	2.0e-02	2.0e-03	2.1
	2.0380e-05	1.0e-02	1.0e-03	7.0
	5.1133e-06	5.0e-03	5.0e-04	25.8

Table: Comparison of first-order (explicit Euler) and second-order (predictor-corrector) time-stepping schemes.

More efficient second-order

The predictor-corrector scheme requires twice as many function evaluations (applications of “filter” or sparse matrix operations).
Once the first time-step y_1 has been computed via predictor-corrector, we can switch to the leap-frog (a.k.a. Verlet) scheme:

$$y_{i+1} = y_{i-1} + 2\Delta t\ f(y_i)) \tag{4}$$

Exercise [BPS85]:

- implement the leap-frog scheme and
- compare it with predictor-corrector with respct to accuracy and efficiency.

Analysis of the difference method

We have combined two centered, second-order differences, one for d/dx and the other for d^2/dx^2. So we expect the combination to be second order. However, there is a representation that makes this very clear and also leads the way to a remarkable fourth-order scheme.

We can write the solution to

$$u_t + f(u)_x - u_{xxt} = 0$$

as

$$u_t = -K * f(u) \tag{5}$$

where the "$*$" denotes convolution and

$$K(x) = -sign(x)\frac{1}{2}e^{-|x|}$$

Convolution representation

The equation $u_t + f(u)_x - u_{xxt} = 0$ can be written as a convolution $u_t = -K * f(u)$ because the Green's function for $1 - d^2/dx^2$ is

$$G(x) = \tfrac{1}{2} e^{-|x|}.$$

Note that

$$K(x) = G'(x) \tag{6}$$

is positive for $x < 0$ and negative for $x > 0$.
We can now approximate u_t via the trapezoidal rule ($h = \Delta x$)

$$u_t(t, ih) \approx -h \sum_{j \neq i} K((j-i)h)\, f(u(t, jh)) \tag{7}$$

Convolution approximation

Using the formulation

$$u_t(t, ih) \approx -h \sum_{j \neq i} K((j-i)h)\, f(u(t, jh))$$

would involve $\mathcal{O}(N^2)$ work, where $N = L/h$ is the number of spatial grid points (L is the length of the computational domain). Note that we can write

$$\begin{aligned} u_t(t, ih) \approx & -h \sum_{j<i} K^-((j-i)h)\, f(u(t, jh)) \\ & -h \sum_{j>i} K^+((j-i)h)\, f(u(t, jh)) \end{aligned} \tag{8}$$

where $K^{\pm} = \mp \frac{1}{2} e^{-|x|}$ since $K^-(0) + K^+(0) = 0$.

Convolution approximation

Write the corresponding algorithm ($v \approx u_t$) as

$$v_i = \sum_{j \in \mathbb{Z}} k_{j-i} f_j \tag{9}$$

where $f_j = f(u(t, jh))$ and $k_\ell = \frac{1}{2} h \operatorname{sign}(\ell) e^{-|\ell| h}$ for $\ell \neq 0$, $k_0 = 0$.

Define the difference operator D^2 via the filter

b=[0 1], a=[0 1 0] + $1/(e^h - 2 + e^{-h})$ [-1 2 -1]

Then $D^2 K^{\pm} = 0$ and $(D^2 k)_i = 0$ for $|i| > 1$ and

$$(D^2 k)_i = \begin{cases} \frac{1}{2} h\, i\, e^{-h} + \frac{h\, i \left(-e^{-2h} + 2e^{-h}\right)}{2(e^h - 2 + e^{-h})} & |i| = 1 \\ 0 & i = 0 \end{cases}$$

Note that $i = \operatorname{sign}(i)$ when $|i| = 1$.

Convolution simplification

Note that

$$
\begin{aligned}
(D^2 k)_{\pm 1} &= \pm \tfrac{1}{2} h\, e^{-h} + \frac{\pm h \left(-e^{-2h} + 2e^{-h}\right)}{2(e^h - 2 + e^{-h})} \\
&= \pm h \frac{e^{-h}(e^h - 2 + e^{-h}) + \left(-e^{-2h} + 2e^{-h}\right)}{2(e^h - 2 + e^{-h})} \\
&= \pm h \frac{(1 - 2e^{-h} + e^{-2h}) + \left(-e^{-2h} + 2e^{-h}\right)}{2(e^h - 2 + e^{-h})} \\
&= \frac{\pm h}{2(e^h - 2 + e^{-h})} = \frac{\pm h}{2(h^2 + h^4/12 + \cdots)} \\
&= \frac{\pm 1}{2h} \left(1 - h^2/12 + \cdots\right).
\end{aligned}
$$

Convolution simplification

Therefore we can write

$$D^2 v = D^1 f$$

where $v = k * f$ as defined in (9) and

$$(D^1 f)_i = \frac{h}{2(e^h - 2 + e^{-h})}(f_{i+1} - f_{i-1})$$

Therefore a 2-nd order scheme is obtained via

$$u_t \approx v$$

where $D^2 v = D^1 f$. Note that

$$e^h - 2 + e^{-h} = h^2 \left(1 + \frac{h^2}{12} + \frac{h^4}{360} + \cdots\right)$$

Gregory-Euler-Maclaurin scheme

We can compute the integral in

$$u_t = -K * f(u)$$

more accurately via a quadrature related to the Euler-Maclaurin formula:

$$\int_0^\infty g(x)\,dx \approx \tfrac{1}{2}hg(0) + h\sum_{i=1}^{\infty} g(ih) + \frac{h^2}{12}g'(0) \quad (10)$$

This scheme is 4th order accurate [Sco11] provided g goes to zero rapidly enough at ∞.
By approximating the derivative via a difference, we obtain a scheme known to Gregory.

Application to convolution

Applying this to the convolution integral we get

$$\begin{aligned} &\int_{-\infty}^{\infty} K(x) f(kh - x)\, dx \\ &\approx h \sum_{i \neq 0} K(ih) f((k-i)h) + \frac{h^2}{12} [(Kf)'] \end{aligned} \tag{11}$$

where $[g] = g(0+) - g(0-)$. We have

$$[(Kf)'] = [K'f + Kf'] = [K'f] + [Kf'] = [K]\, f'(0) = -f'(0).$$

We can retain a 4th order scheme by approximating $f'(0)$ to second order:

$$f'(0) \approx \frac{f(h) - f(-h)}{2h}$$

Convolution simplification

Define

$$(Jf)_i = \frac{h}{24}(f(i+1) - f(i-1))$$

Therefore a 4th order scheme is obtained via

$$u_t \approx v^1 + v^2$$

where

$$D^2 v^1 = D^1 f$$

and

$$v^2 = Jf$$

Note that

$$J = \frac{e^h - 2 + e^{-h}}{12} D^1.$$

Convolution simplification

Using the 4th order GEM scheme together with a 4th order Runge-Kutta scheme for time stepping, we get an overall 4th order scheme.

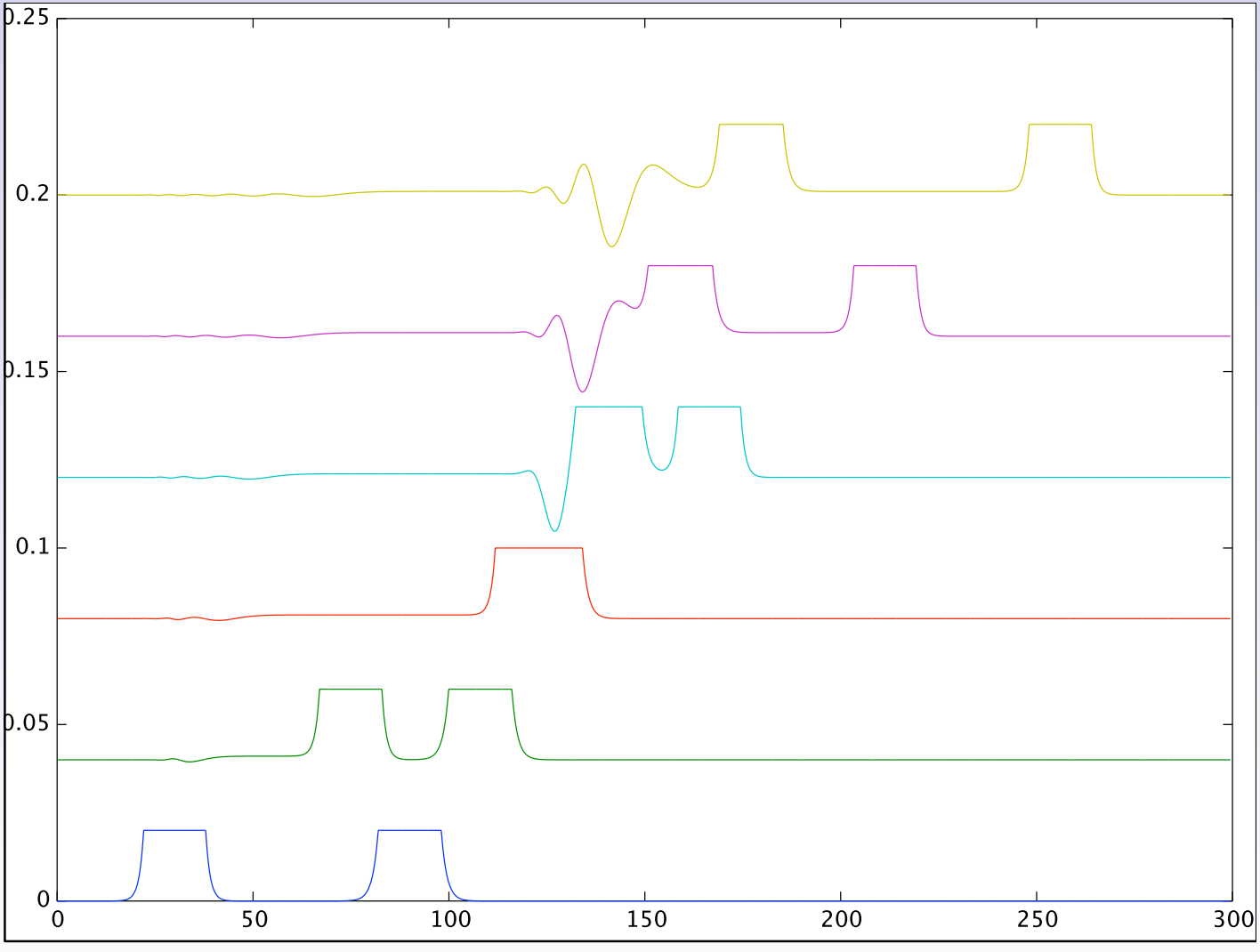
We compare this with the 2nd order approximation for the BBM equation using predictor-corrector (2nd order) time stepping for propagating a soliton of amplitute 3 for 30 time units.

	rel. max err.	dx	dt	cpu secs
predictor	1.8e-03	5.0e-02	5.0e-03	4.2
corrector	4.7e-04	2.5e-02	2.5e-03	14.9
fourth	1.7e-03	1.0e-01	1.0e-01	0.46
order	7.0e-05	1.0e-01	5.0e-02	0.73
	3.7e-09	1.0e-02	5.0e-03	39.1
	2.7e-10	5.0e-03	2.5e-03	177.0
	9.1e-10	2.0e-03	1.0e-03	1056.9

Table: Comparison of second-order scheme and 4th-order scheme.

Solitary interaction detail

Solitary interaction for BBM is not exact.



Solitary interaction details

As reported in [BPS80], the solitary interaction for BBM produces an oscillatory tail of maximum amplitude -0.01674.

[BPS80] noted that the larger solitary wave increases in amplitude due to the interaction.

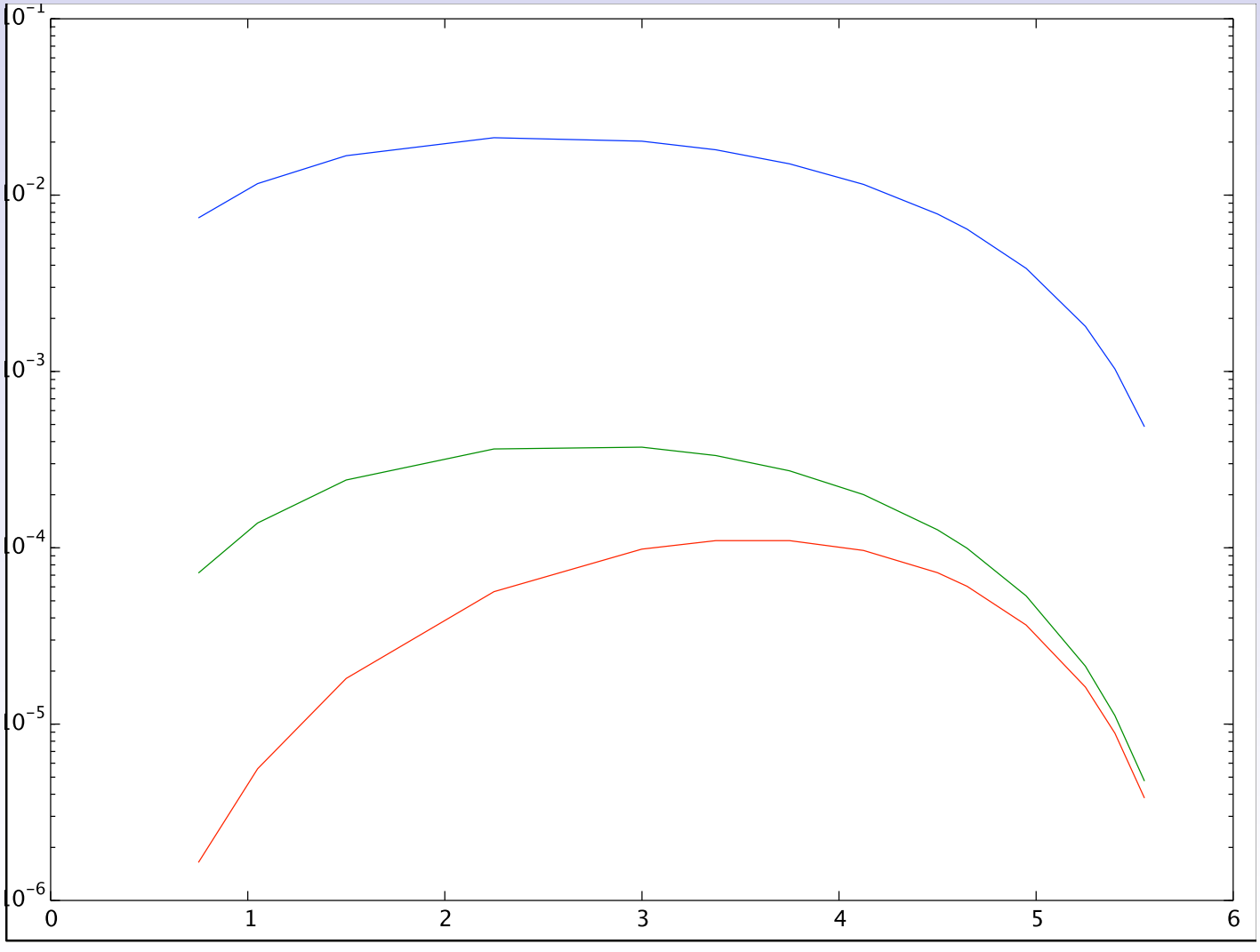
We find that the larger wave has increased in size by 1.822×10^{-05}, only 3.037×10^{-06} times the original amplitude.

In [BPS80], the increase in size was reported to be over two orders of magnitude larger.

The smaller solitary wave decreases in amplitude by 2.422×10^{-04}, in agreement with the results reported in [BPS80].

Nonexact interaction

The top curve is the amplitude of the dispersive tail, middle curve is the decrease in the smaller wail, and the bottom curve is the increase in the larger wave.



The 4-th order Runge-Kutta method

The standard Runge-Kutta method used is closely related to Simpson's rule:

$$
\begin{aligned}
w^1 &= u^n - \tfrac{1}{2}\Delta t F(u^n) \\
w^2 &= u^n - \tfrac{1}{2}\Delta t F(w^1) \\
w^3 &= u^n - \Delta t F(w^2) \\
u^{n+1} &= u^n - \frac{\Delta t}{6}\Big(F(u^n) \\
&\quad + 2F(w^1) + 2F(w^2) + F(w^3)\Big)
\end{aligned}
\tag{12}
$$

This has four function evaluations per time step.

A 5-th order method

A fifth-order method can be obtained by combining 4-th order Adams-Bashforth (as a predictor)

$$\hat{u}^{n+1} = u^n - \frac{\Delta t}{24}\Big(55F(u^n) - 59F(u^{n-1}) + 37F(u^{n-2}) - 9F(u^{n-3})\Big) \tag{13}$$

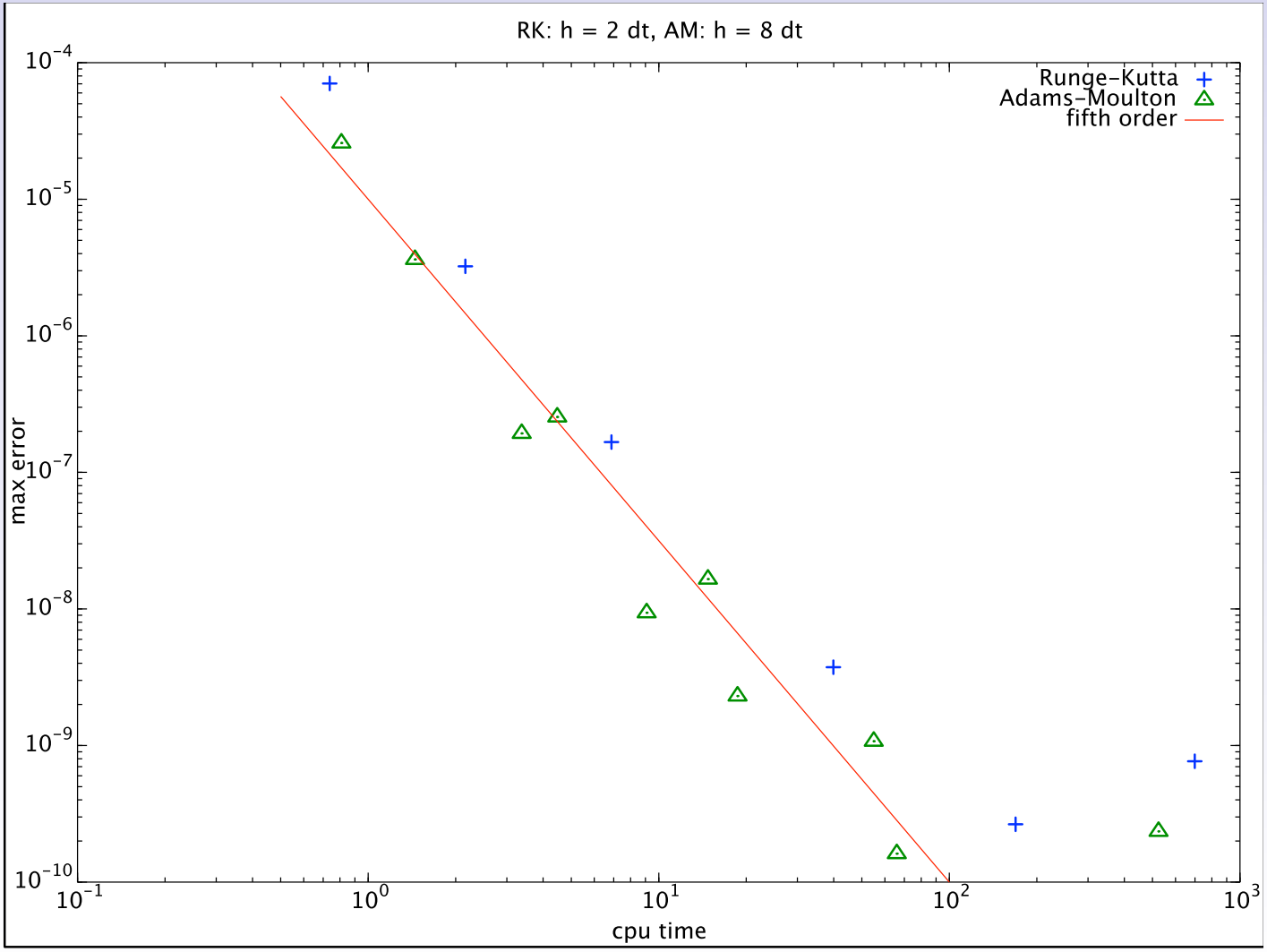
with 5-th order Adams-Moulton (implicit normally)

$$u^{n+1} = u^n - \frac{\Delta t}{720}\Big(251F(\hat{u}^{n+1}) + 646F(u^n) - 264F(u^{n-1}) + 106F(u^{n-2}) - 19F(u^{n-3})\Big) \tag{14}$$

This has two function evaluations per time step.

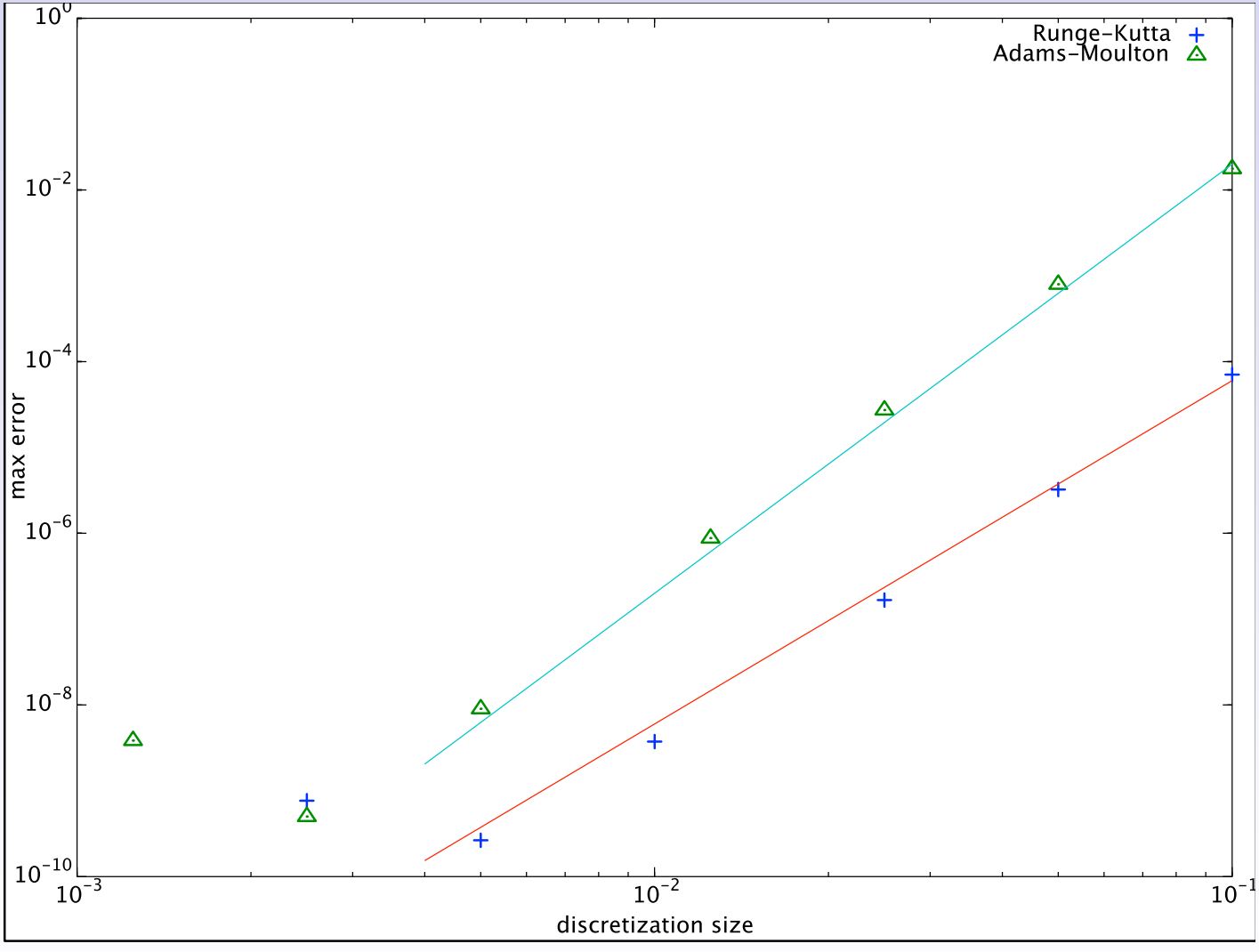
RK versus AM

Solitary wave propagation test for two time-stepping methods: error versus cpu time. For Runge-Kutta, $h = 2\Delta t$. For Adams-Moulton, $h/\Delta t$ takes values between 2 and 8.



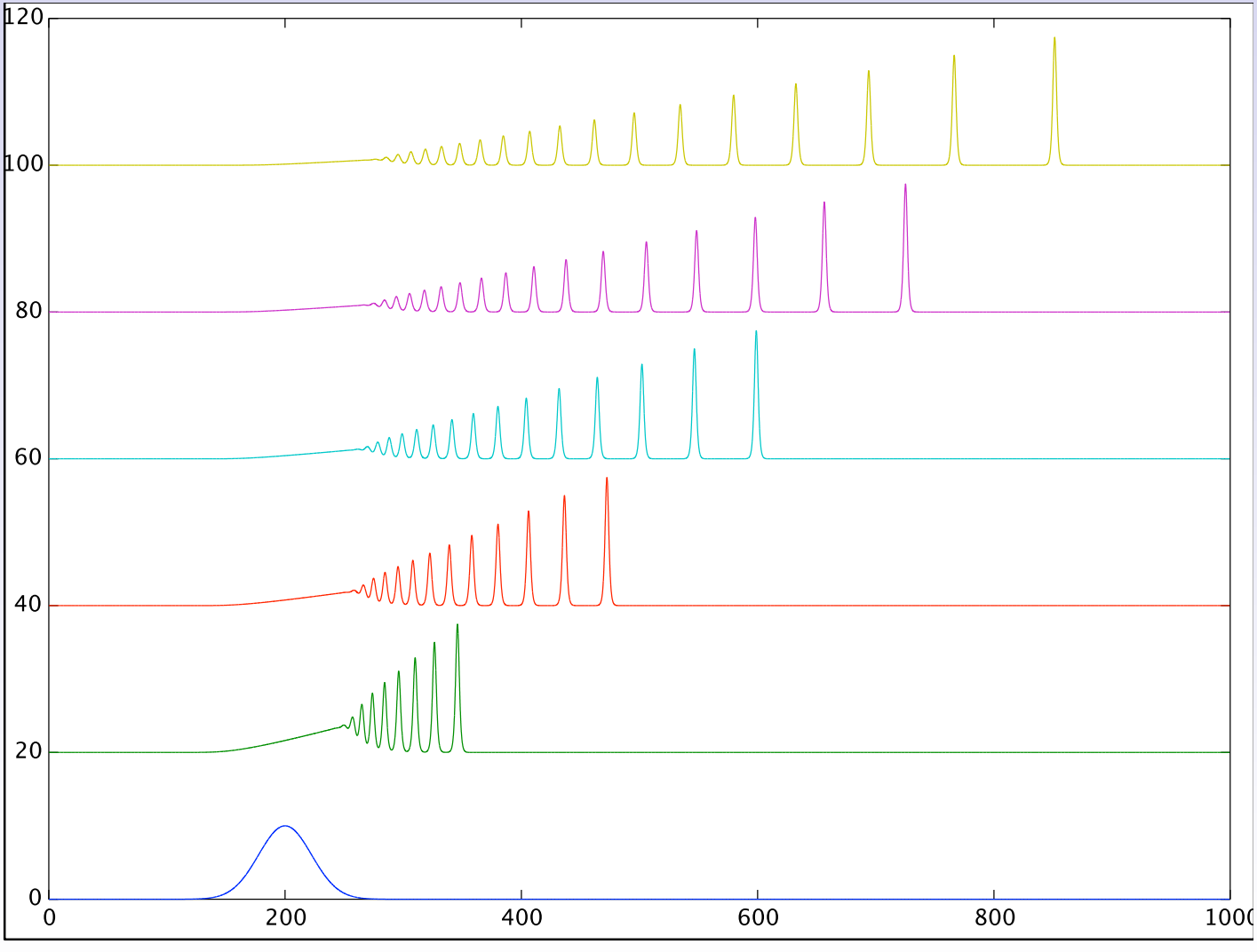
RK versus AM

Solitary wave propagation test for two time-stepping methods: error versus mesh size for $h = \Delta t$.

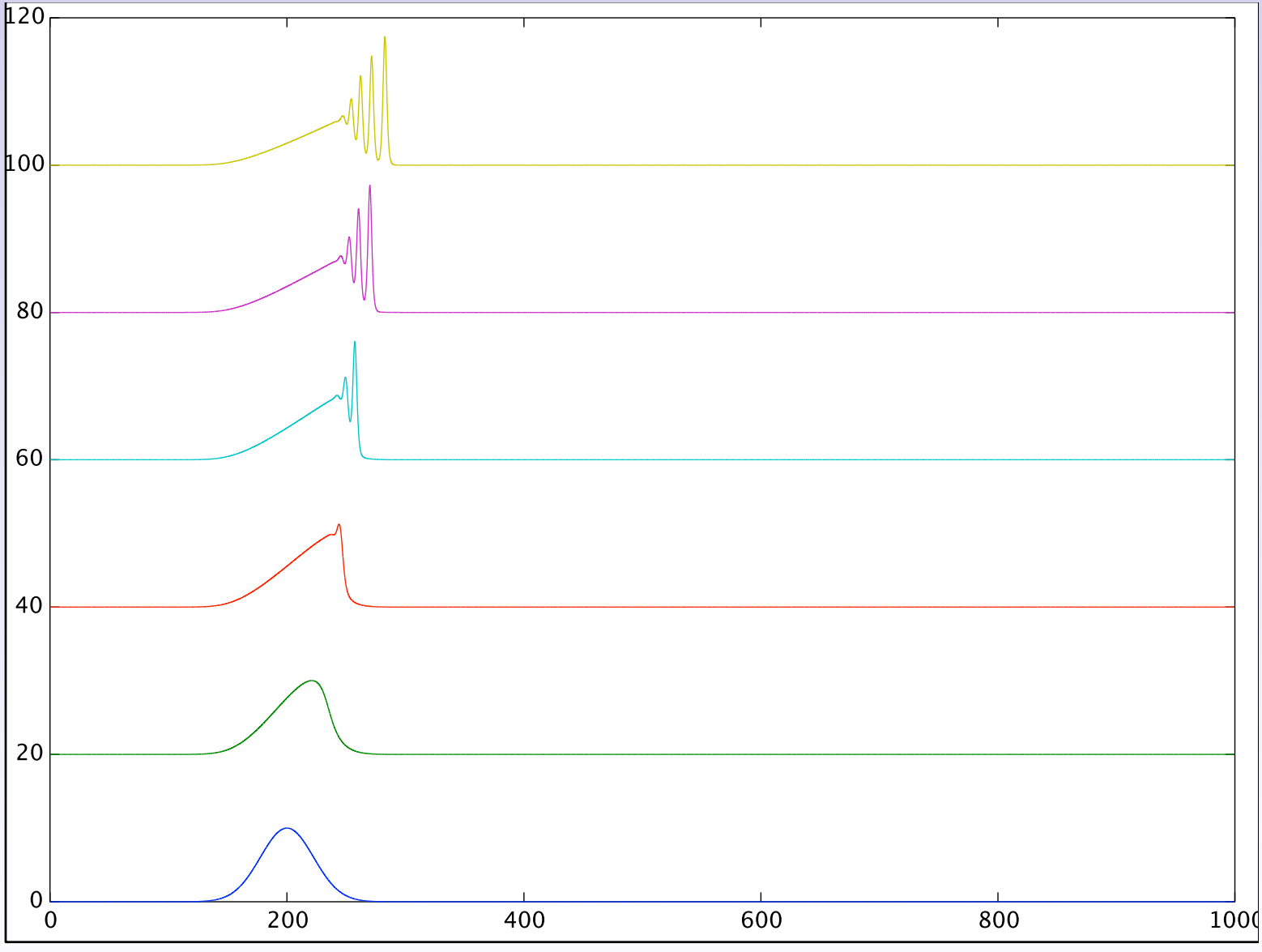


New things

Efficient schemes allow the exploration of new phenomena [HS74]



Close up of big Gaussian



Boussinesq using FEniCS

The FEniCS Project automates simulation code [LMW12].

The Boussinesq equations have been implemented using FEniCS tools, in particular using Dolfin [LPT12].

FEniCS tools include GPU support [Rat10].

The FEniCS book [LMW12] can be obtained at http://fenicsproject.org/book/index.html

See “Download from Launchpad”

J. L. Bona, W. G. Pritchard, and L. R. Scott, *Solitary-wave interaction*, Physics of Fluids **23** (1980), 438–441.

_____, *An evaluation of a model equation for water waves*, Philos. Trans. Roy. Soc. London Ser. A 302 (1981), 457–510.

_____, *Numerical schemes for a model for nonlinear dispersive waves*, J. Comp Phys. **60** (1985), 167–186.

J.L. Hammack and H. Segur, *The Korteweg-de Vries equation and water waves. part 2. comparison with experiments*, Journal of Fluid mechanics **65** (1974), no. 02, 289–314.

A. Logg, K.A. Mardal, and G. Wells, *Automated solution of differential equations by the finite element method: The FEniCS book*, vol. 84, Springer, 2012.

N.D. Lopes, P.J.S. Pereira, and L. Trabucho, *Improved boussinesq equations for surface water waves*, Automated Solution of Differential Equations by the Finite Element Method: T he FEniCS Book (A. Logg, K.A. Mardal, and G. Wells, eds.), Springer-Verlag New York Inc, 2012, pp. 471–504.

F. Rathgeber, *Automated finite element computations in the fenics framework using general purpose graphics processing units*, Ph.D. thesis, 2010.

L. Ridgway Scott, *Numerical analysis*, Princeton Univ. Press, 2011.