2018 Summer Undergraduate Research Project

UTSD equation & Hyperdiffusion equaiton

Khang Ee Pang
Supervised by Lennon Ó Náraigh & Andrew Gloster

30th July 2018

Background - Triple Point Paradox

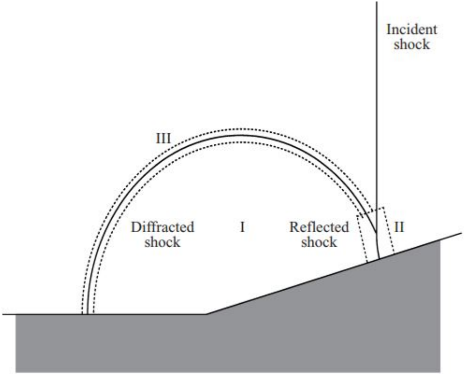


Figure: Triple point of weak shock reflection off a thin wedge. Image obtained from Hunter and Brio (2000).

Background - Triple Point Paradox

J. Fluid Mech. (2000), *vol.* 410, *pp.* 235–261. Printed in the United Kingdom 235

Weak shock reflection

By JOHN K. HUNTER[1] AND MOYSEY BRIO[2]

[1]Department of Mathematics and Institute of Theoretical Dynamics, University of California, Davis, CA 95616, USA

[2]Department of Mathematics, University of Arizona, Tucson, AZ 85721, USA

Hunter and Brio provide a numerical solutions for what now known as the UTSD equation. They suggest that there is a supersonic patch behind the triple point which resolves the paradox.

UTSD Equation

The Unsteady Transonic Small-Disturbance (UTSD) equation is used to describe the shock structure when a sufficiently weak shock reflects off a wedge.

$$\frac{\partial^2 u}{\partial x \partial t} + \frac{\partial^2 F}{\partial x^2} + \frac{\partial^2 u}{\partial y^2} = 0$$

Discretization

we use a spatial discretization of (5.4) of the form

$$u_{i,j}^{n+1} - \sigma(u_{i,j+1}^{n+1} - 2u_{i,j}^{n+1} + u_{i,j-1}^{n+1})$$
$$= u_{i+1,j}^{n+1} + u_{i,j}^{n} - u_{i+1,j}^{n} + \nu(f_{i+3/2,j}^{n} - f_{i+1/2,j}^{n} - f_{i-1/2,j}^{n} + f_{i-3/2,j}^{n}), \quad (5.5)$$

where

$$\nu = \frac{\Delta t}{\Delta x}, \qquad \sigma = \frac{\Delta t \Delta x}{\Delta y^2}.$$

Discretization

we use a spatial discretization of (5.4) of the form

$$u_{i,j}^{n+1} - \sigma(u_{i,j+1}^{n+1} - 2u_{i,j}^{n+1} + u_{i,j-1}^{n+1})$$
$$= u_{i+1,j}^{n+1} + u_{i,j}^{n} - u_{i+1,j}^{n} + \nu(f_{i+3/2,j}^{n} - f_{i+1/2,j}^{n} - f_{i-1/2,j}^{n} + f_{i-3/2,j}^{n}), \quad (5.5)$$

where

$$\nu = \frac{\Delta t}{\Delta x}, \qquad \sigma = \frac{\Delta t \Delta x}{\Delta y^2}.$$

Does not support parallelization.

Discretization

Discretize UTSD in time (same apporach)

$$\frac{\partial}{\partial x}\left(\frac{u^{n+1}-u^{n}}{\Delta t}\right)+\frac{\partial^2 F^n}{\partial x^2}+\frac{\partial^2 u^{n+1}}{\partial y^2}=0$$

Discretization

Discretize UTSD in time (same apporach)

$$\frac{\partial}{\partial x}\left(\frac{u^{n+1}-u^n}{\Delta t}\right)+\frac{\partial^2 F^n}{\partial x^2}+\frac{\partial^2 u^{n+1}}{\partial y^2}=0$$

Re-arrange

$$\frac{\partial u^{n+1}}{\partial x}+\Delta t\frac{\partial^2 u^{n+1}}{\partial y^2}=Q^n(x,y), \qquad Q^n(x,y)=\frac{\partial u^n}{\partial x}-\Delta t\frac{\partial^2 F^n}{\partial x^2}$$

Discretization

Discretize UTSD in time (same apporach)

$$\frac{\partial}{\partial x}\left(\frac{u^{n+1}-u^n}{\Delta t}\right)+\frac{\partial^2 F^n}{\partial x^2}+\frac{\partial^2 u^{n+1}}{\partial y^2}=0$$

Re-arrange

$$\frac{\partial u^{n+1}}{\partial x}+\Delta t\frac{\partial^2 u^{n+1}}{\partial y^2}=Q^n(x,y), \qquad Q^n(x,y)=\frac{\partial u^n}{\partial x}-\Delta t\frac{\partial^2 F^n}{\partial x^2}$$

We need to solve a backwards heat equation for every timestep.

Backwards heat equation

Solving this with Fourier transform (with the appropriate initial and boundary condition) we get

$$\psi(x,y) = \sum_{n=0}^{\infty} \widehat{\psi}_n(x)\Psi_n(y) + b(x)f(y)$$

where

$$\widehat{\psi}_n(x) = \widehat{\psi}_n(L)e^{-\kappa k_n^2(L-x)} - \int_x^L e^{-\kappa k_n^2(x'-x)}\widehat{q}_n(x')dx'$$

and $\Psi_n(y)$ is the orthogonal function. And support parallelization.

UTSD equation

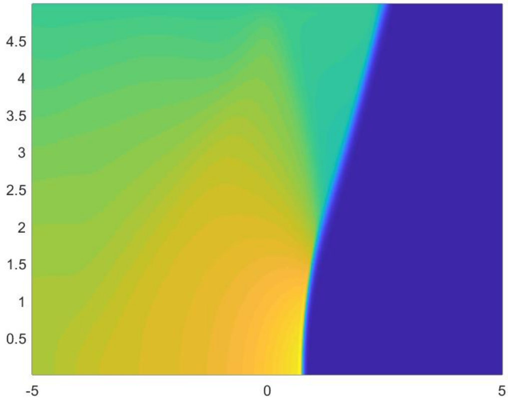


Figure: Neumerical simulation after 400 timestep.

Discrete Cosine Transform

Recall the backwards heat equation

$$\frac{\partial u^{n+1}}{\partial x} + \Delta t \frac{\partial^2 u^{n+1}}{\partial y^2} = Q^n(x, y)$$

We need to transform $Q^n(x, y)$ at each timestep.

Discrete Cosine Transform

Recall the backwards heat equation

$$\frac{\partial u^{n+1}}{\partial x} + \Delta t \frac{\partial^2 u^{n+1}}{\partial y^2} = Q^n(x, y)$$

We need to transform $Q^n(x, y)$ at each timestep.
Which is given by

$$X_k = \sum_{n=0}^{N-1} x_n \cos\left[\frac{\pi}{N}\left(n + \frac{1}{2}\right)\left(k + \frac{1}{2}\right)\right], \qquad k = 0, 1, 2, ..., N-1$$

for each row of $Q^n(x, y)$.

Discrete Cosine Transform

Original approach:

- Matrix multiplication
- Storage size: N
- Operation count: $O(N^2)$

Discrete Cosine Transform

Original approach:

- Matrix multiplication
- Storage size: N
- Operation count: $O(N^2)$

Alternative approach:

- Fast Cosine Transform (FCT)
- Storage size: $2\sqrt{N}$
- Operation count: $O(N \log_2 N)$

Discerte Cosine Transform

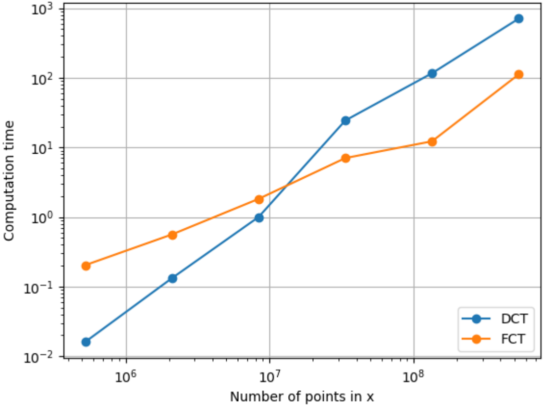


Figure: Speed comparison of DCT and FCT.

Conclusion

Conclusion

- Support parallel computing
- Implement Fast Cosine Transform
- Improve accuracy: uses analytic solution*
- Produce MATLAB code for above implimentation

Further improvement

- Better approach to resolve discontinunity

Background

Cahn-Hilliard equation

$$\frac{\partial C}{\partial t} = D\nabla^2(C^3 - C - \gamma\nabla^2 C), \qquad t > 0$$

Application in polimer physics and interfacial flows.

Background

Cahn-Hilliard equation

$$\frac{\partial C}{\partial t} = D\nabla^2(C^3 - C - \gamma\nabla^2 C), \qquad t > 0$$

Application in polimer physics and interfacial flows.
This boils down to solving the hyperdiffusion equation

$$\frac{\partial C}{\partial t} = -\gamma D\nabla^4 C, \qquad t > 0$$

Periodic pentadiagonal matrix

Crank-Nicholson scheme

$$\left(\begin{array}{ccccccc|cc}
c & d & e & 0 & & & 0 & b & a \\
b & c & d & e & 0 & & & 0 & b \\
a & b & c & d & e & 0 & & & 0 \\
0 & a & b & c & d & e & 0 & & \\
 & \ddots & \ddots & \ddots & \ddots & \ddots & \ddots & \ddots & \\
 & & 0 & a & b & c & d & e & 0 \\
0 & & & 0 & a & b & c & d & e \\
\hline
e & 0 & & & 0 & a & b & c & d \\
d & e & 0 & & & 0 & a & b & c
\end{array}\right)
\begin{pmatrix} C_1^{n+1} \\ C_2^{n+1} \\ C_3^{n+1} \\ C_4^{n+1} \\ \vdots \\ C_{N-3}^{n+1} \\ C_{N-2}^{n+1} \\ C_{N-1}^{n+1} \\ C_N^{n+1} \end{pmatrix}
=
\begin{pmatrix} d_1^n \\ d_2^n \\ d_3^n \\ d_4^n \\ \vdots \\ d_{N-3}^n \\ d_{N-2}^n \\ d_{N-1}^n \\ d_N^n \end{pmatrix}$$

LU factorization

For a generic equation

$$Ax = b$$

LU factorization

For a generic equation

$$Ax = b$$

We factorize A into $L \times U$

$$LUx = b$$

where L is lower triangular and U is upper triangular matrix.

LU factorization

For a generic equation

$$Ax = b$$

We factorize A into $L \times U$

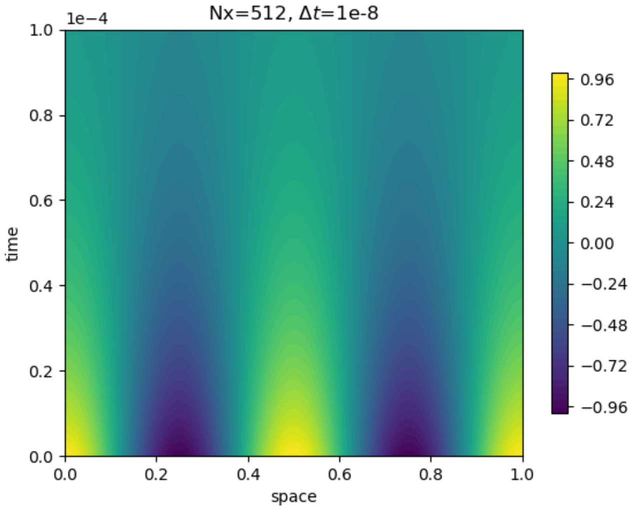
$$LUx = b$$

where L is lower triangular and U is upper triangular matrix. Use forward and backward substitution to invert L and U respectively

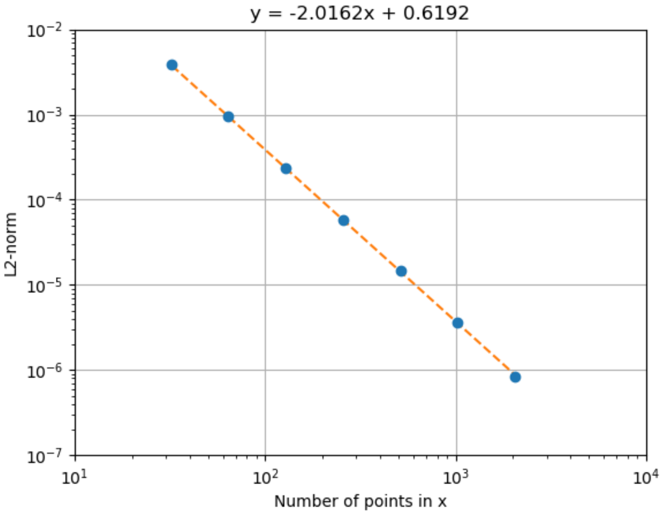
$$Ux = L^{-1}b$$

$$x = U^{-1}L^{-1}b$$

Hyperdiffusion equation



Convergence analysis



2D Hyperdiffusion equation

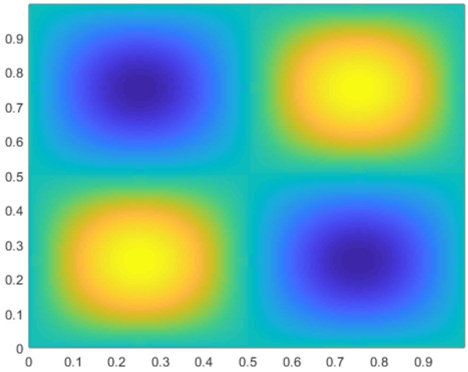


Figure: Neumerical simulation after 2000 timestep

Conclusion

- Improve algorithm for solving the hyperdiffusion equation
- Produce serial code in C to solve 1D hyperdiffusion equation in batch
- Produce serial code in C to solve 2D hyperdiffusion equation

Reference

I J. K. Hunter and M. Brio. Weak shock reflection. *J. Fluid Mech.*, 2000.

II L. Ó Náraigh. New idea for the UTSD equation. 2018.

III L. Ó Náraigh. New idea for Parallelizing the 1D Heat Equation. 2018.

IV L. Ó Náraigh and A. Gloster. Potential applications for a pentadiagonal solver. 2018.

V I. M. Navon. Pent: A periodic pentadiagonal solver. *Communications in Applied Numerical Methods*, 1987.

VI A. Gloster and L. Ó Náraigh. cuPentBatch - A Batch Pentadiagonal Solver for NVIDIA GPUs. 2018.