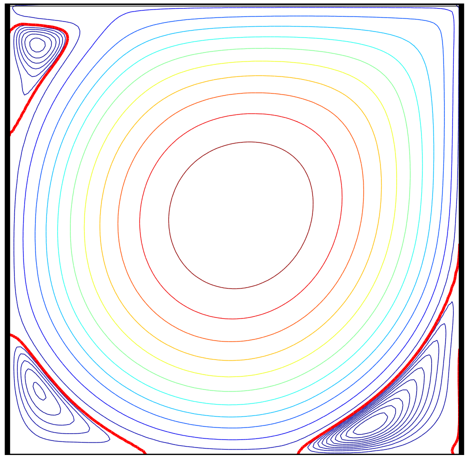
Grid-Coupling for the Lattice-Boltzmann-Method

Martin Rheinländer

Le may s'écoule si rapide
et cette fois même trop vite.
Quand je voulais jouir de la vie
hélas, il s'était déjà fini.



Doktoranden-Seminar
2nd June 2003

Outline

- goal: Account of my Activity
- introduction: context around LBM
- scaling: numerical experiment and analysis
- presentation of software
- 2D benchmarks
- grid coupling
- numerical tests

There will be also many colored pictures ;-) !

Introduction to LBM

Purpose: Numerical solution of the *incompressible Navier-Stokes* equation,

$$\begin{aligned} \partial_t \mathbf{u} + \mathbf{u}\cdot\nabla\mathbf{u} &= -\nabla p + \nu\Delta\mathbf{u} \\ \operatorname{div}\mathbf{u} &= 0 \\ \text{initial condition:} \quad & \mathbf{u}|_{t=0} = \mathbf{u}_0 \\ \text{boundary condition:} \quad & \mathbf{u}|_{\partial\Omega} = \ldots \end{aligned}$$

describing the motion of fluids.

How it works: The LBM is an explicit finite difference scheme, given by *The (discrete) Lattice-Boltzmann Equation*, with a *relaxation term* on the l.h.s.

$$F_k(\mathbf{x} + \delta\mathbf{x}_k, t + \delta t) = F_k(\mathbf{x}, t) - \tfrac{1}{\tau}\Big(F_k(\mathbf{x}, t) - \mathcal{E}_k\big(r(\mathbf{x}, t), \mathbf{u}(\mathbf{x}, t)\big)\Big)$$

$$\mathbf{u} = c\textstyle\sum_k F_k\mathbf{e}_k, \quad r = 1 + 3c^{-2}p = \sum_k F_k, \quad c = \delta x/\delta t \quad \delta\mathbf{x}_k = \delta x\mathbf{e}_k$$

The primary quantities F_k are called *populations*.

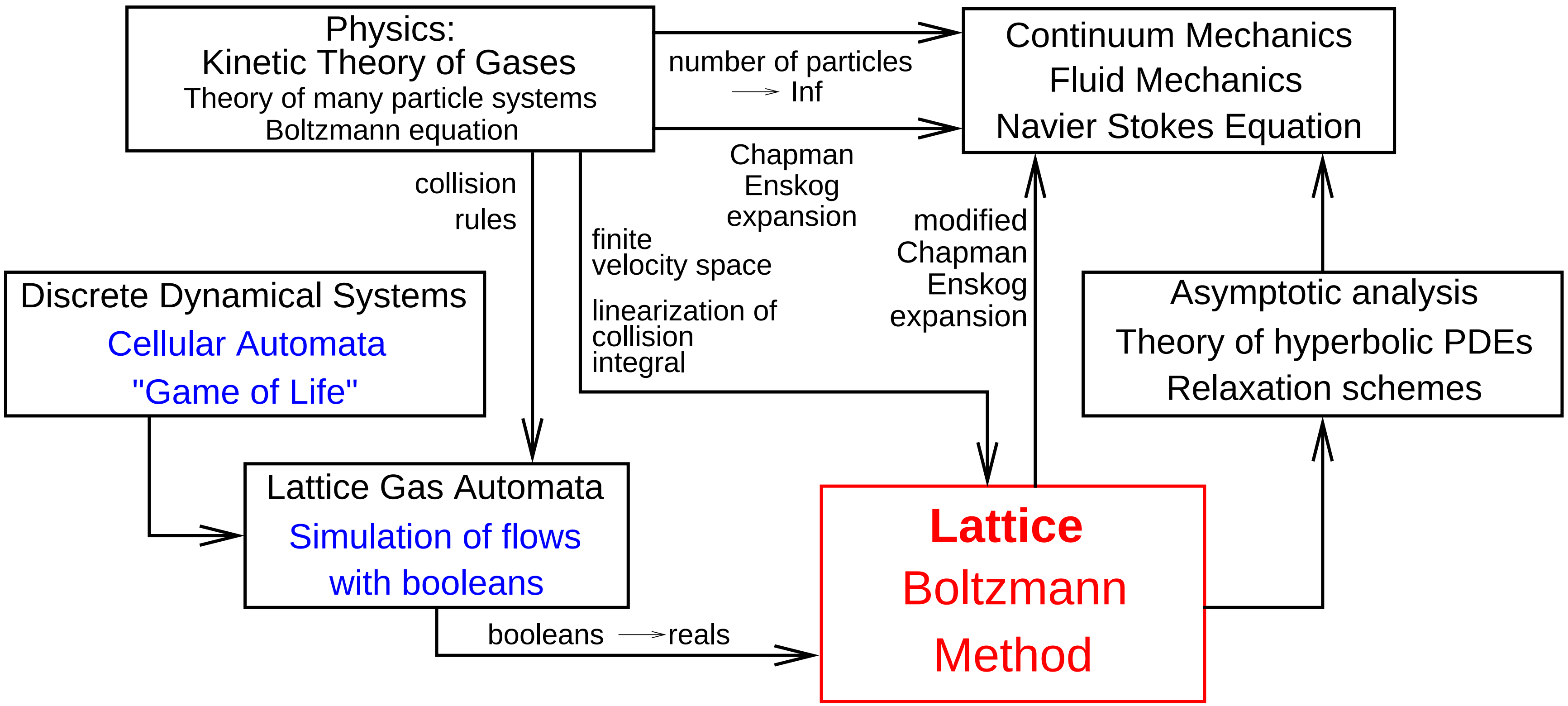
Background of LBM

Idea of relaxation schemes:

+ simplify structure of equation
– increase number of variables

Non-Linearity $\leftrightarrow$ *High Dimension*

History:



Overview: Comparison of Discretization Methods

- *Finite Elements*: equation $\longrightarrow$ discretization
  - variational formulation based on well-defined function spaces
  - find finite-dimensional *approximating* function spaces
  - consider restricted problem

- *Finite Differences*: equation $\longleftrightarrow$ discretization

$\rightarrow$ replace derivatives by suitable difference quotients
$\leftarrow$ perform consistency analysis

- *Lattice-Boltzmann Method*
  NS equation $\overset{i)}{\longleftarrow}$ continuous LBE (ϵ) $\overset{ii)}{\longleftrightarrow}$ discrete LBE $(\delta s(\epsilon), \delta t(\epsilon))$

i) For $\epsilon \downarrow 0$: Convergence of derived quantities to solution of NS equation.

ii) Classical LBM: two simultaneous limit processes in sequence of refinements. $\Rightarrow$ *FD notion of consistency* for discrete LBE w.r.t. continuous LBE makes no sense.

Different Scalings for LBM

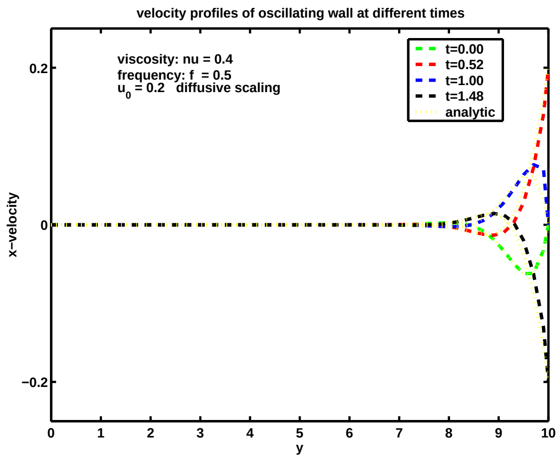
Relation between collision time τ and viscosity $\nu = \frac{2\tau-1}{6}\frac{\delta s^2}{\delta t}$

acoustic scaling	diffusive scaling
$\delta s = \delta t \quad \Rightarrow \quad c := \frac{\delta s}{\delta t} = 1 \quad \nu = \frac{2\tau-1}{6}\delta s$	$\delta s = \delta t^2 \quad \Rightarrow \quad c := \frac{\delta s}{\delta t} = \frac{1}{\delta s} \quad \nu = \frac{2\tau-1}{6}$

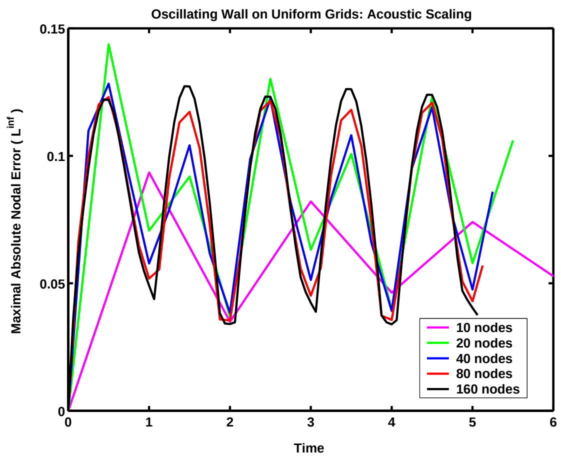
Example of differing behavior, where acoustic scaling fails:

$$\partial_t u - \nu \partial_{xx} u = 0 \qquad u(0,t) = 0; \qquad u(H,t) = u_0 \sin(\omega t)$$

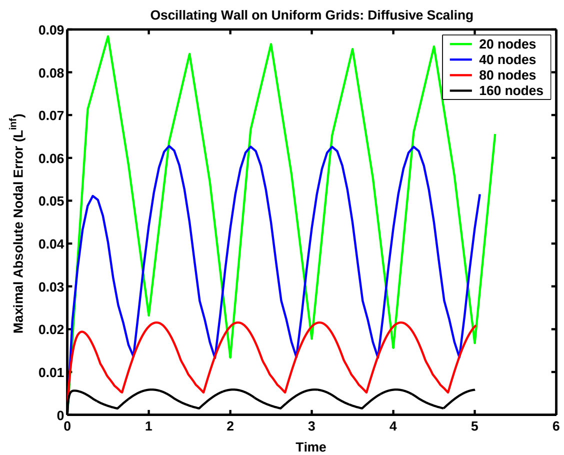
$$u(x,t) = u_0 \Im\left(\frac{\sinh((1+i)\gamma x)}{\sinh((1+i)\gamma H)} e^{i\omega t}\right) \qquad \gamma := \sqrt{\omega/2\nu}$$



several flow profiles



acoustic scaling: l^∞ error



diffusive scaling: l^∞ error

Reduction of Populations for quasi 1D Flows

- *parallel shear flows:* translation-invariant in x-direction
  2D incompressible Navier-Stokes eqn. $\longrightarrow$ *1D linear advection-diffusion eqn.*
- reduction of physical quantities: $u, v, p \longrightarrow u$
- Observation:
  - LBGK-algorithm respects translational invariance in the direction of coordinate-axes
  - simulation of *parralel shear flows* needs only linear part of equilibrium

$$\mathcal{E}_k(r, \mathbf{u}) = \underbrace{w_k \big[r + 3c^{-2}\mathbf{u}\cdot\mathbf{c}_k}_{\mathcal{E}_{k,\mathrm{lin}}(r,u,v)} + \frac{9}{2}c^{-4}(\mathbf{u}\cdot\mathbf{c}_k)^2 - \frac{3}{2}c^{-2}\mathbf{u}^2\big]$$

Theorem: Initialize D2Q9-LBGK algorithm by: $F_k(x, y, 0) = \mathcal{E}_{k,\mathrm{lin}}(1, \bar{u}(y), 0)$

$\Rightarrow$ 9 population scheme $\longrightarrow$ 3 population scheme

Continous LB Equations for 3 population scheme

reduced scheme with *signs* $(s_k)_k = (0, -1, 1)$, *weights* $(w_k)_k = (\frac{2}{3}, \frac{1}{6}, \frac{1}{6})$:

$$\begin{aligned} P_k(y + s_k\delta y, t + \delta t) - P_k(y,t) &= -1/\tau \left(P_k(y,t) - Q_k(y,t)\right) \\ u := P_0 + P_1 + P_2 \quad \text{equilibrium: } Q_k &:= w_k\, u + 1/2\, s_k\, c^{-1}\, a\, u \end{aligned}$$

- Question: How to relate this *"discrete"* FD equation to a *"continuous"* PDE ?
- consider: $\partial_t p_k(y,t) \; + \; \beta \; s_k \; \partial_y p_k(y,t) = - \; \gamma \; \frac{1}{\tau} \; \left(p_k(y,t) - q_k(y,t)\right)$
- integrating along characteristics $t' \mapsto (y + s_k \beta t', t + t')$ from t to $t + \delta t$ and approximating integral of relaxation term by *left side rule*:

$$p_k(y + s_k\beta\delta t + \delta t) - p_k(y,t) = - \; \gamma \; \delta t \; \frac{1}{\tau}\left(p_k(y,t) - q_k(y,t)\right)$$

- thus, the FD eqn. is recovered, if: $\beta\delta t = \delta y$ and $\gamma = \frac{1}{\delta t}$.

Analysis of Acoustic and Diffusive Scaling

- *acoustic scaling:* $\frac{\delta y}{\delta t} = \beta = 1, \qquad \delta t = \gamma^{-1} = \epsilon$

$$\partial_t p_k(y,t) \; + \; s_k \, \partial_y p_k(y,t) = - \, \epsilon \tau^{-1} \left(p_k(y,t) - q_k(y,t)\right) \tag{1}$$

- *diffusive scaling:* $\frac{\delta y}{\delta t} = \beta = \epsilon, \qquad \delta t = \gamma^{-1} = \epsilon^2$

$$\partial_t p_k(y,t) \; + \; \epsilon \, s_k \, \partial_y p_k(y,t) = - \, \epsilon^2 \tau^{-1} \left(p_k(y,t) - q_k(y,t)\right) \tag{2}$$

- introduction of moments:

acoustic scaling	diffusive scaling
$m_0^a = p_0 + p1 + p2$	$m_0^d = p_0 + p1 + p2$
$m_1^a = p_2 - p1$	$m_1^d = \epsilon^{-1}(p_2 - p1)$
$m_2^a = p_2 + p1$	$m_2^d = p_2 + p1$

Asymptotic Behavior

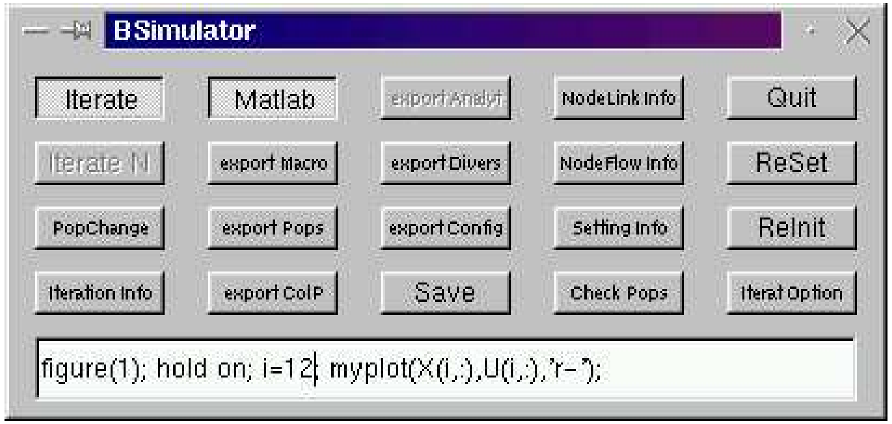
- algebraic manipulations: *continous LBE* $\rightarrow$ *equivalent moment system*
- assumption: moments remain bounded for $\epsilon \downarrow 0$, i.e. are $O(1)-$function.
- combination of these new equations $\Rightarrow$ *macroscopic limit equations:*

acoustic scaling	diffusive scaling
$\partial_t u + a\partial_y u - \frac{\tau\epsilon}{3}\partial_{yy} u - \tau\epsilon\partial_t m_1 = O(\epsilon^2)$	$\partial_t u + a\partial_y u - \frac{\tau}{3}\partial_{yy} u = O(\epsilon^2)$

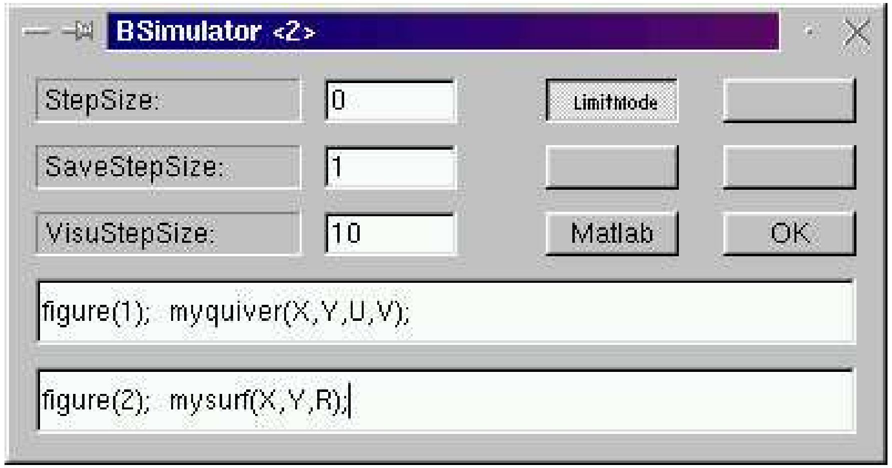
Software Development

- since january 2003: development of a 2D LB code in *C++*
  - basic program based on uniform square grids
  - extension for refinement zones (patches)
  - interpolation tool
- purpose:
  - demonstration and validation of the LB method (numerical experiments)
  - further developments, e.g. boundary conditions, grid coupling
- requirements:
  - simulation of standard benchmarks,
  - direct visualization, which is very desirable for time-dependent problems
  - changing parameters without recompiling
  - usable also for other people (hopefully)

Program Features



CommandWindow



OptionWindow

- integration of MatLab by command lines $\Rightarrow$ direct visualization and analysis of output data (manually resp. automatically)
- interactive communication between program and user even if the computation is running
- possibility of saving and retrieving data

Technical Ingredients

- object oriented design: implemetation in $C{+}{+}$ $\Rightarrow$ modularity
- Qt based graphical user interface (GUIs/windows)
- use of POSIX threads $\Rightarrow$ feigns simultaneity of computation and interaction with the user on a single processor machine
- use of the MatLab Application Interface in order to start MatLab from the program as postprocessing tool
- one–way coupling: program sends data to Matlab via a pipe

The program is still 50% terminal based, i.e. program and MatLab output is displayed on the parent terminal, also the terminal serves as input window where it is possible, in order to loose not too much time in the GUI programming.

Thread_2_Function

Messenger ↑
MatlabEngine
MatlabInterface
while(*Quit*=false)
  switch(*Command2*)
    getInfo
    export data
    send MatlabCommand
    start/stop Matlab
  default: sleep
end
clean up & exit

main

init *QApplication, Messenger,*
  CommandWindow,
  OptionWindow
set SIGNAL–SLOT connections
  CommandWindow ↔ OptionWindow
  CommandWindow → Quit
init *Mesh (Messenger)*
SPLIT PROGRAM
  by calling thread functions
show CommandWindow
enter Qt–event loop
  controlling the GUIs

Thread_1_Function

Messenger ↑
MatlabEngine
MatlabInterface
while(*Quit*=false)
  switch(*Command1*)
    ITERATE(*Messenger*↑)
    resetMesh(*Messenger*↑)
    reinitMesh(*Messenger*↑)
    start/stop Matlab
  default: sleep
end
clean up & exit

CommandWindow

Messenger↑

OptionWindow

Messenger↑

Interface Messenger

Mesh ↑
bool *Command1, Command2,*
  Quit
int *Thread1_ID, Thread2_ID*

Mesh

discretization
LBM algorithm

Class Design for LBM Algorithm

Global Aspects

- elementary geometric classes:
  `class TGrid; class TRectangle;`
- definition of configuration:
  `class TConfig;`
- assembly of mesh patches is defined in the node type array of `class TPatch;`
  initialization by elementary classes
- `class TSubMesh`: assembles node array links nodes, global evolution methods, inherits `TPatch`
- `class TMesh`: links interface nodes of submeshes, global time-step method
- `class TMatlabInterf`: collects data from nodes in matrices, sends them to MatLab

Local Aspects

```
class TNode{
   double Population[9];
   double ColProduct[9];
   TNode* Neighbour[9];
   void propagate();
   void collide();
};
```

derived node classes with special propagation methods:

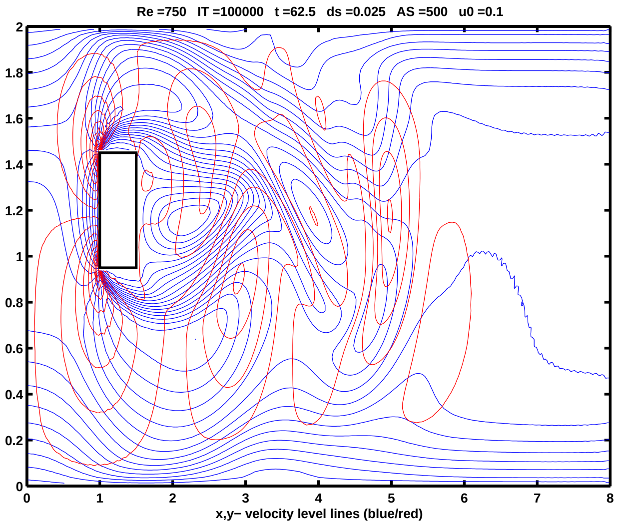
`class TLidNode;`
`class TEdgeNode;`
`class TInletNode;`
`class TOutletNode;`
	boundary node types

`class TInterfNode;`
`class TCoupleNode;`
`class THangingNode;`
	coupling node types

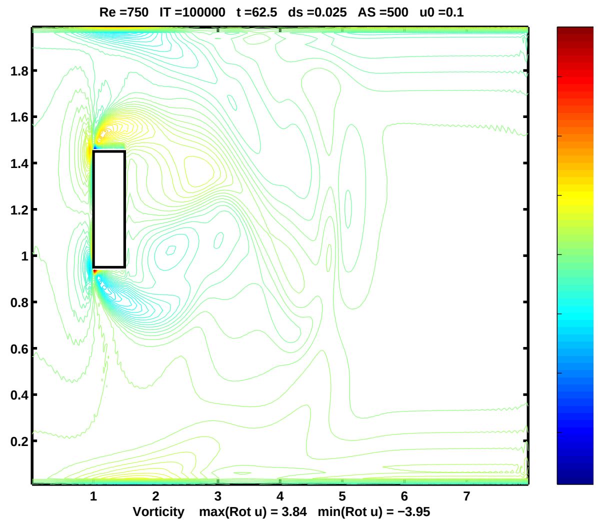
Standard 2D Benchmarks

1. Flow around an Asymmetrically Placed Rectangle

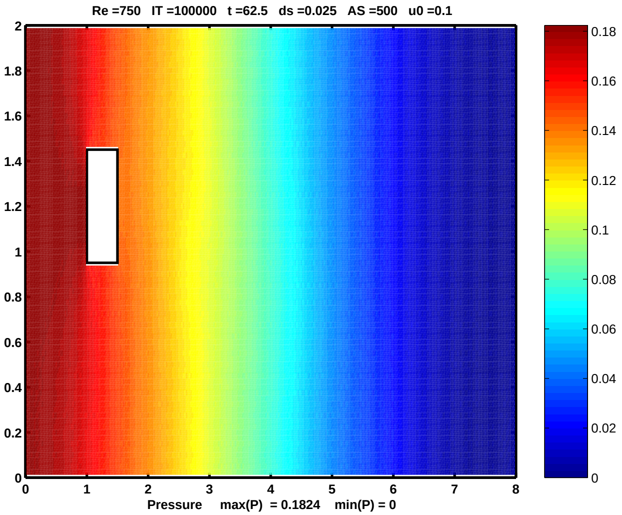
$Re = 750, \;\; 80 \times 320$ **LBGK**



x,y-velocity



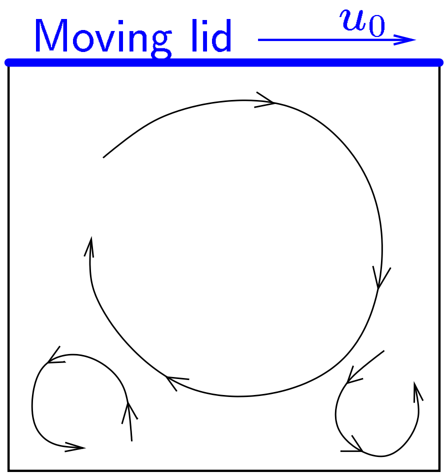
Vorticity

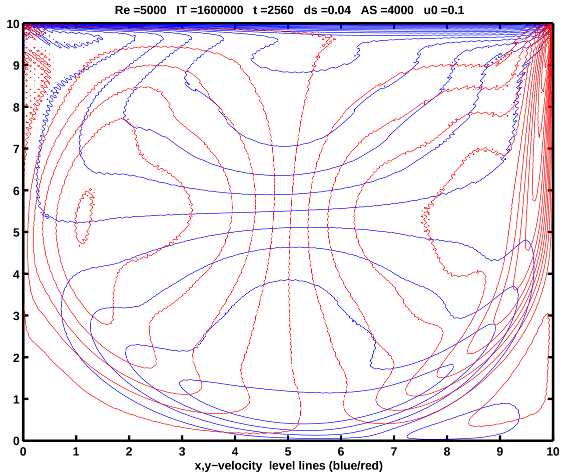


Pressure

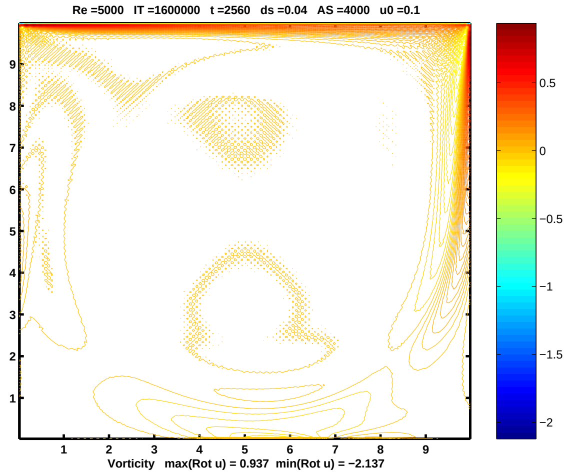
“Dream”: full developed Karmàn vortex street

2. Lid Driven Cavity $Re = 5000, \;\; 250 \times 250$ **LBGK**



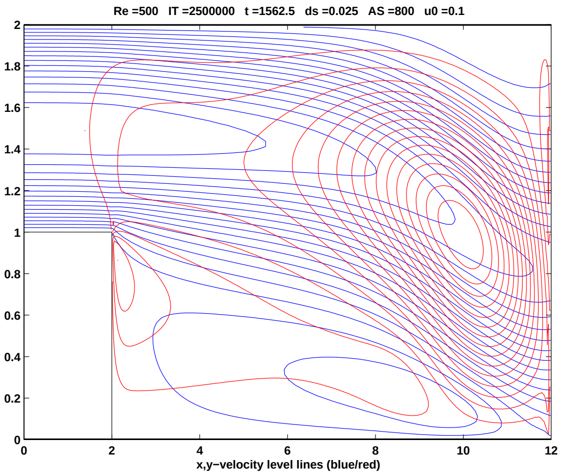


x,y-velocity

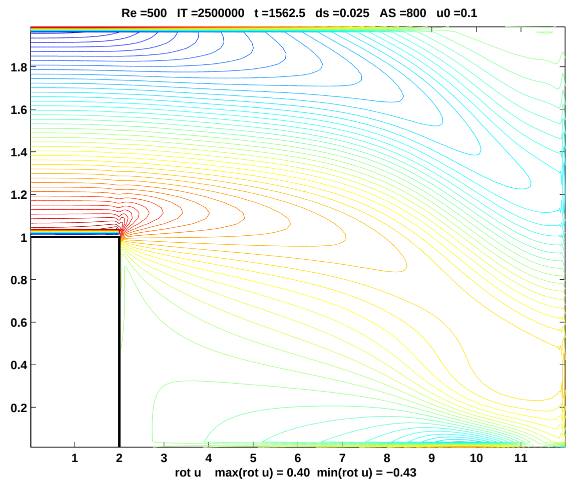


Vorticity

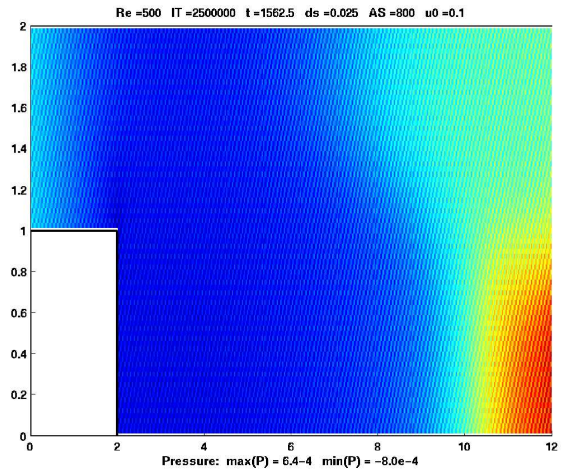
3. Flow over Backward Facing Step $Re = 500 \;\; 80 \times 480$ **LBGK**



x,y-velocity



Vorticity



Pressure

Simulation of Standard 2D Benchmarks: Remarks

- boundary conditions:
  - *Dirichlet b.c.* for moving lid and inlet boundaries: *bounce-back* + correction term for non-zero boundary velocity
  - emulation of the the *no-slip* boundary condition ⇐ comes out naturally by a variational ansatz for NS equation [lecture notes R.Rannacher]

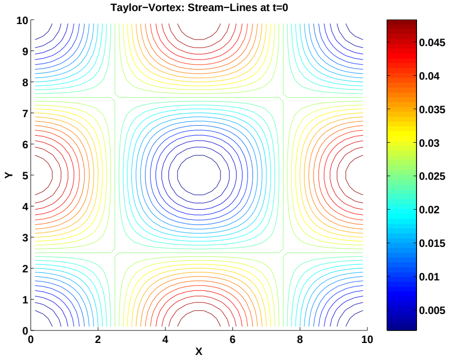
$$\nu\partial_{\mathbf{n}}\mathbf{u} - p\mathbf{n} = 0 \text{ at outlet}, \quad \int_{outlet} p = 0$$

translation into LB: equilibrium + velocity of left neighbor + pressure set to 0
too restrictive, not consistent (?!) and not satisfactory à la longue

- how to check accuracy and reliability in space and time without any external help?
  - 1st idea: perform computation with equal parameter setting (u_0, ν,,geometry) on a sequence of refined grids → interpolation tool to compare results on different grids
  - 2nd idea: perform computations with equal Reynolds numbers but different parameter settings → compare corresponding times
- problems with pressure, different behavior of vorticity and divergence, although both are computed by derivatives of velocity field (?)

2D Bechmarks with Analytic Solution

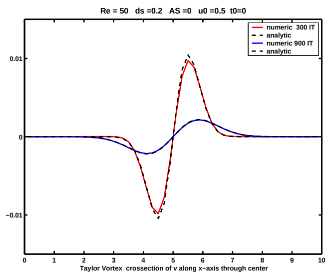
Periodic Vortex:

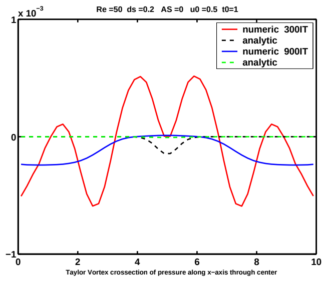


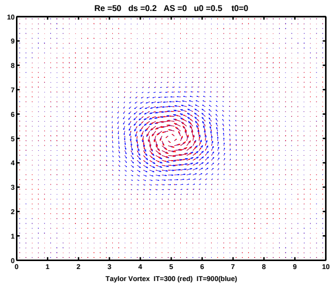
$$
\begin{aligned}
u(x,y,t) &= -\frac{u_0}{a}\cos(ax)\,\sin(by)\exp\left(-\nu(a^2+b^2)t\right)\\
v(x,y,t) &= \frac{u_0}{b}\sin(ax)\,\cos(by)\exp\left(-\nu(a^2+b^2)t\right)\\
p_{\text{Navier}}(x,y,t) &= -\frac{u_0^2}{4}\left(\frac{\cos(2ax)}{a^2}+\frac{\cos(2by)}{b^2}\right)\exp\left(-2\nu(a^2+b^2)t\right)\\
p_{\text{Stokes}}(x,y,t) &= 0
\end{aligned}
$$

Taylor Vortex (rotationally sym.)

$$
\begin{aligned}
\mathbf{u}(r,t) &= \frac{r}{\nu\ t^2}\exp\left(-r^2\big/4\nu t\right)\mathbf{e}_\phi\\
p_{\text{Navier}}(r,t) &= -\frac{1}{\nu\ t^3}\exp\left(-r^2\big/2\nu t\right)\\
p_{\text{Stokes}}(r,t) &= 0
\end{aligned}
$$







Remark on initial conditions

2D Bechmarks with Analytic Solution: Remarks

- problems with pressure (depending on parameters)
  Note: if (almost) mass conservation holds for the LB algorithm, then the mean pressure w.r.t. computation area remains constant, whereas the real pressure may change its average value (e.g. Taylor vortex as solution in $\mathbb{R}^2$).

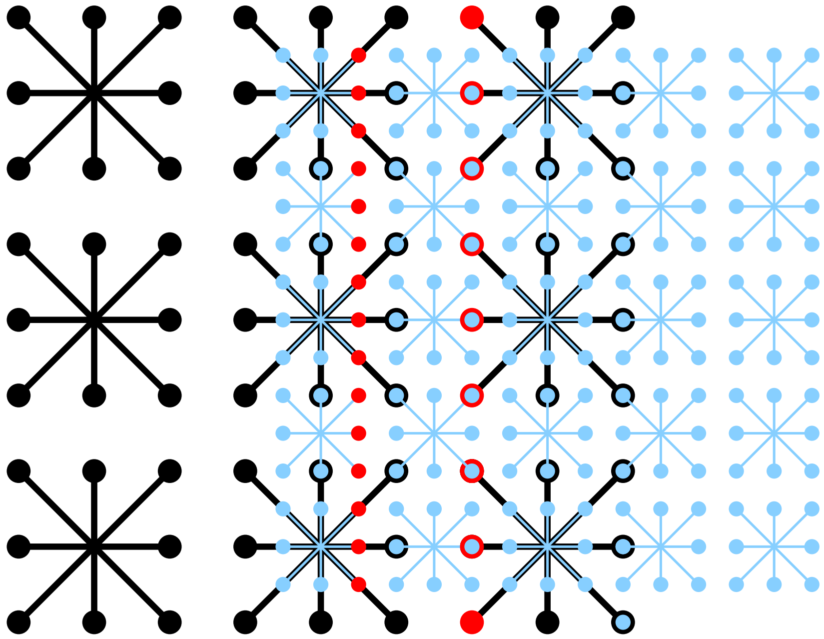
- vortex solutions are not representative enough benchmarks
  they have as solutions of Stokes and Navier-Stokes equation only diffusive character
  $\Rightarrow$ phenomena due to advection cannot be observed.

Property *(Galilei invariance of the NS equation):* Let the pair $(\mathbf{u}(\mathbf{x},t), p(\mathbf{x},t))$ be a solution of the incompressible NS equation and $\mathbf{c}$ a given velocity vector; then the pair

$$\tilde{\mathbf{u}}(\mathbf{x},t) := \mathbf{u}(\mathbf{x}-t\mathbf{c},t) + \mathbf{c} \qquad \tilde{p}(\mathbf{x},t) := p(\mathbf{x}-t\mathbf{c},t)$$

is also a solution of the NS equation (but not Stokes !).

Grid Coupling: The Problem



grid interface with overlap

- problem: fill empty populations (red)

$\Leftrightarrow$ find a rule, that prescribes values for those populations, whose characteristics come out of the the interface (coupling condition)

- difficulty: diffusive scaling entails $c = \frac{1}{\delta s} \Rightarrow$ discontinous grid velocity $\Rightarrow$ jumping populations

- consequence: interpolation of populations due to “discontinuity” must not be applied

- way out: impose continuity conditions of other/physical quantities and recover unknown populations

Grid-Coupling: Heuristic Approaches

à la volume finis:

à la differences finies:

- FD-coupling less complicated than FV-coupling but “not consistent” with half-distance boundary interpretation of bounce-back
- overlapping: *interface node* can get “full information” by its *master-node* ⇒ avoids disturbing effects due to interpolation as much as possible
- two less successful possibilities of coupling:
  - determine unknown populations by solving linear equation system, to assure equality of some moments with *master-node* moments ⇒ which ones ? system might be singular !
  - u, v, p will be prescribed by *master-node*; set equilibrium for either all or only unknown populations ⇒ equilibrium is already bad as boundary condition

Grid-Coupling: Heuristic Approaches (cont.)

- current idea: Find a linear transformation T_{c_1,c_2}, such that

$$T_{c_1,c_2} \cdot \mathcal{E}(r,u,v,c_1) = \mathcal{E}(r,u,v,c_2) \quad \text{and} \quad \langle m(c_2), T_{c_1,c_2} \cdot F \rangle = \langle m(c_1), F \rangle$$

  for any moment generating polynomial m and any population tuple F.

- charcterization: Is there a unique mapping T_{c_1,c_2} with these properties ? This mapping should then be constructed by the polynomials, that are orthogonal w.r.t. to the scalar generated by the LB equilibrium weights.

- What is a moment ? Consider polynomial $p(\mathbf{c}) = p(c_x, c_y)$ and restrict it to the D2Q9 star. ⇒ the associated moment is then:

$$\sum_k p(\mathbf{c}_k) F_k$$

Moment Transformation

$$
\begin{aligned}
m_1(\mathbf{c}) &= 1 && \text{pressure: } \langle m_1, F\rangle = 1 + 3c^{-2}p \\
m_2(\mathbf{c}) &= c_x && \text{x-velocity: } \langle m_2, F\rangle = u \\
m_3(\mathbf{c}) &= c_y && \text{y-velocity: } \langle m_3, F\rangle = v \\
m_4(\mathbf{c}) &= c_x c_y \\
m_5(\mathbf{c}) &= c_x^2 - c_y^2 \\
m_6(\mathbf{c}) &= c_x^2 + c_y^2 - \frac{2}{3}c^2 \\
m_7(\mathbf{c}) &= 3c_x c_y^2 - c^2 c_x \\
m_8(\mathbf{c}) &= 3c_x^2 c_y - c^2 c_y \\
m_9(\mathbf{c}) &= 18c_x^2 c_y^2 - 6(c_x^2 + c_y^2) + 2
\end{aligned}
$$

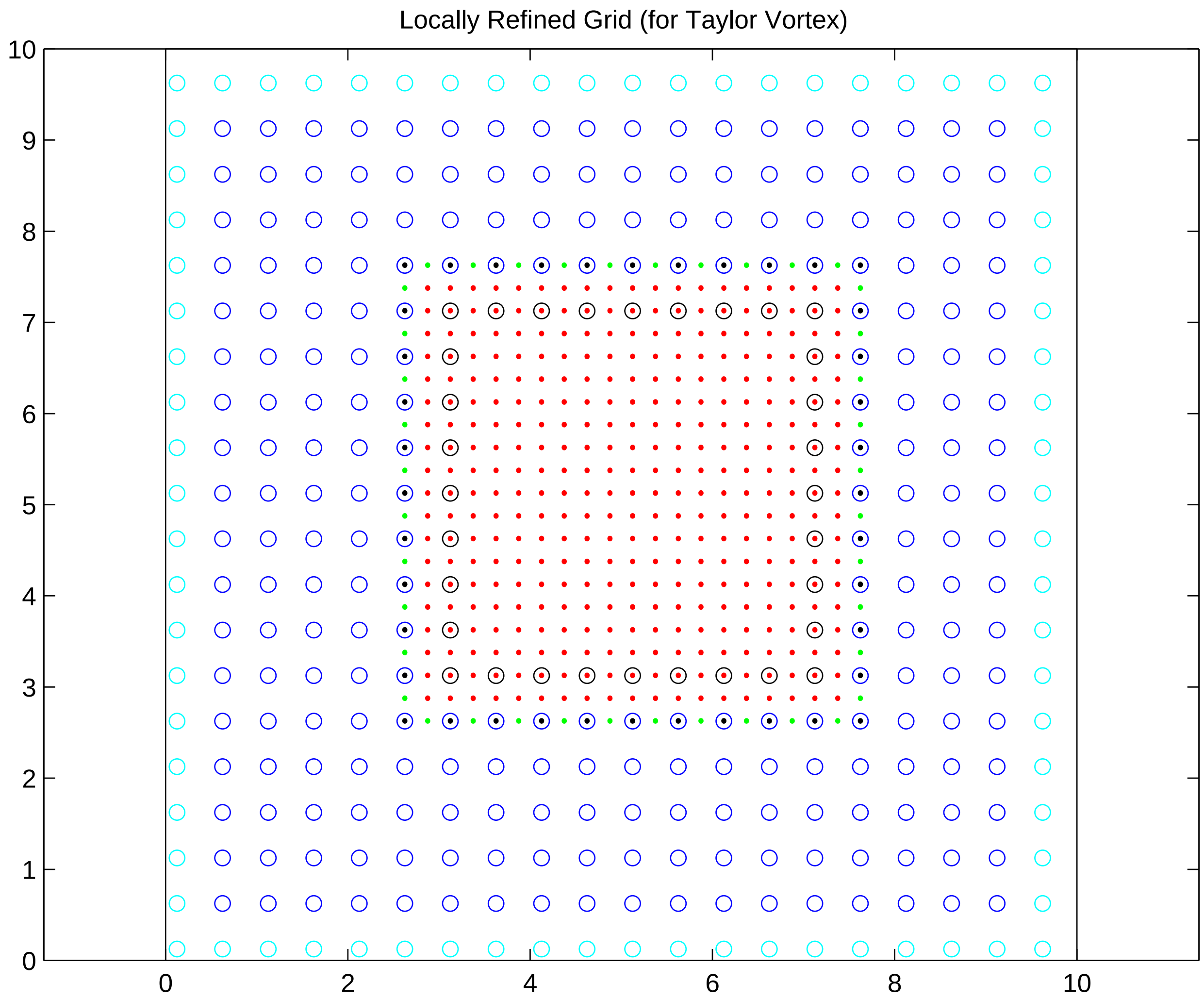
moment transformation:

$$F_B = M^{-1}(1) \cdot C^{-1}(c_B) \cdot C(c_A) \cdot M(1) \cdot F_A$$

modified moment transformation, taking pressure relation into account:

$$F_B = M^{-1}(1) \cdot \Big(1_p + \tilde{C}^{-1}(c_B) \cdot \tilde{C}(c_A) \cdot \big(M(1) \cdot F_A - 1_p\big)\Big)$$

Grid with Refined Patch



Legend:

CIRCLES	=	coarse grid nodes
blue	=	bulk nodes
cyan	=	edge nodes
black	=	interface nodes

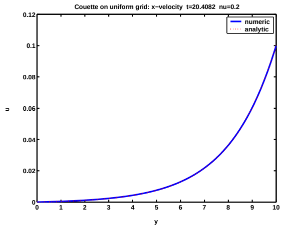
DOTS	=	fine grid nodes
red	=	bulk nodes
black	=	couple nodes
green	=	hanging nodes

Organization of the Time Step

```
void TMesh::doTimeStep(int Level){
  SubMesh[Level]->collide();
  SubMesh[Level]->propagate();
  SubMesh[Level]->advanceTime();

  if(Level<LevelNum-1){
    int RO = getRefOrd(SubMesh[Level],SubMesh[Level+1]);
    SubMesh[Level+1]->getCoupleMoments();
    //recursion
    for(int it=0; it<RO*RO; it++) doTimeStep(Level+1);
    SubMesh[Level]->updateInterface();
  } if(Level==0){t+=dt; IT++;}
}
```

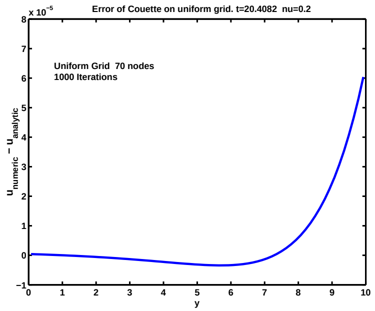
The Role of Interpolation



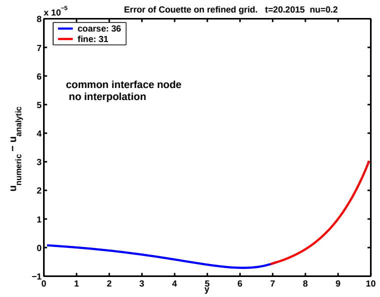
$$v_0\partial_x u - \nu\partial_{xx}u = 0 \quad u(0) = 0 \quad u(H) = u_0$$

$$R := \sqrt{v_0 H/\nu} \quad u(x) = u_0\frac{e^{Ry/H}-1}{e^R-1} \quad \text{(Couette with cross flow)}$$

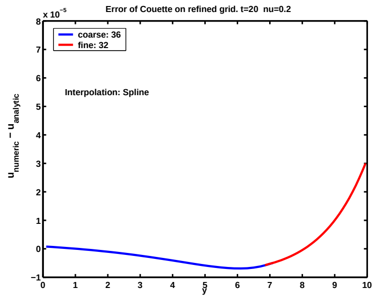
$$\nu = 0.2, \quad v_0 = 0.1, \quad u_0 = 0.1, \quad H = 10$$



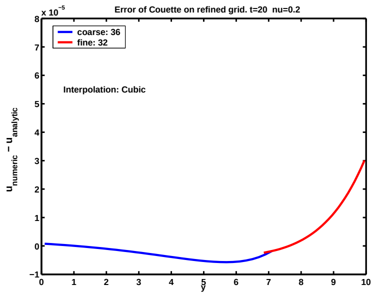
uniform grid



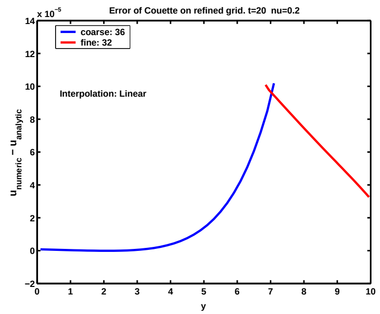
common interface node



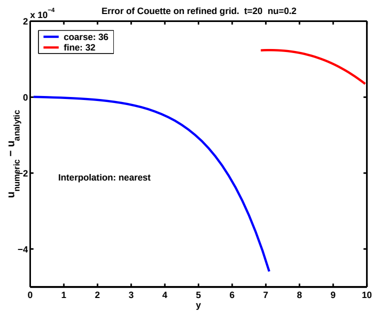
spline interp.



cubic interp.

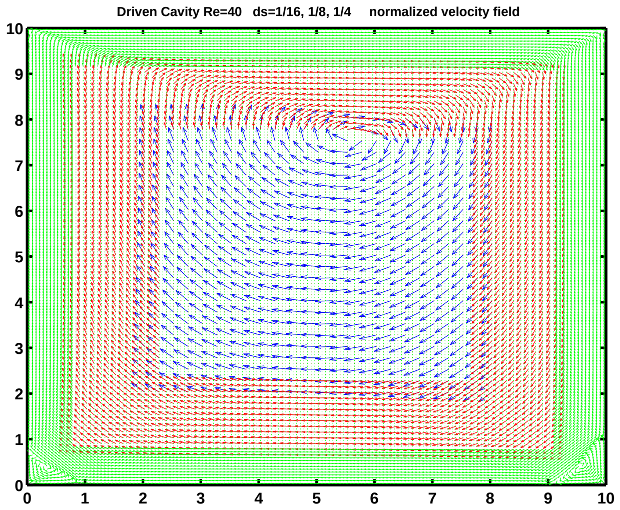


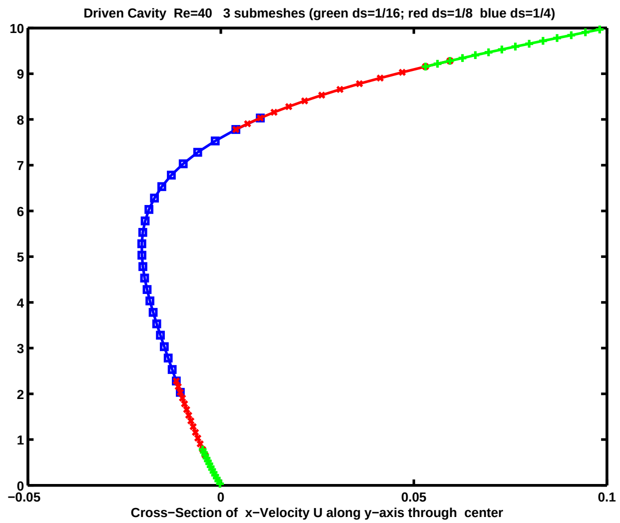
linear interp.

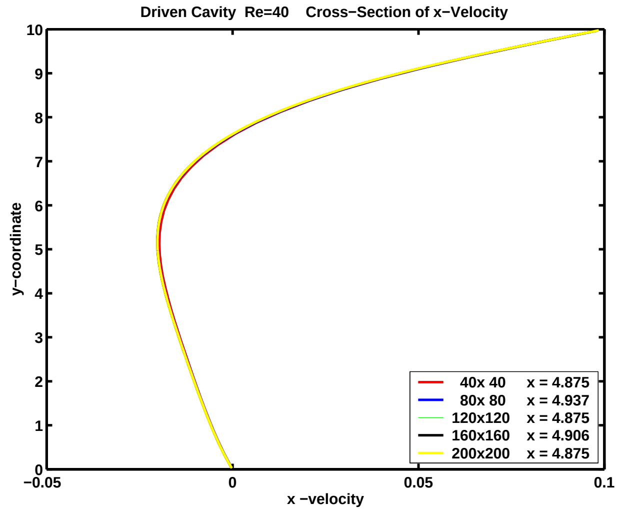


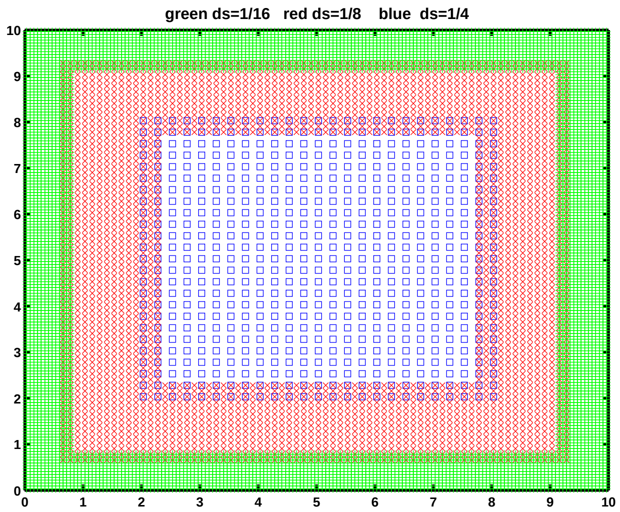
nearest interp.

Lid Driven Cavity $Re = 40$: Refinement around Boundary

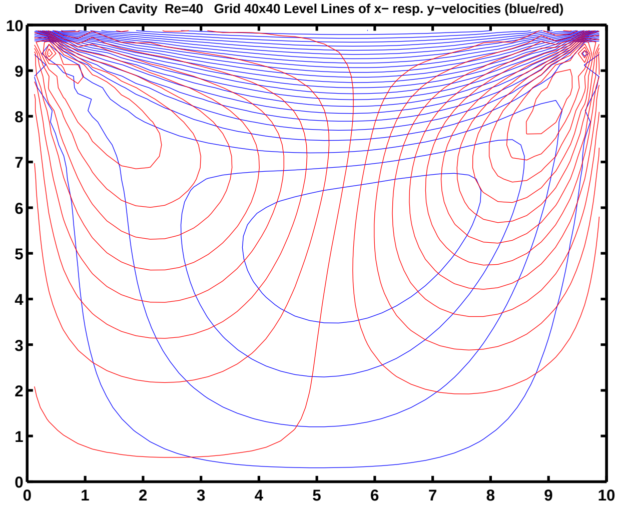


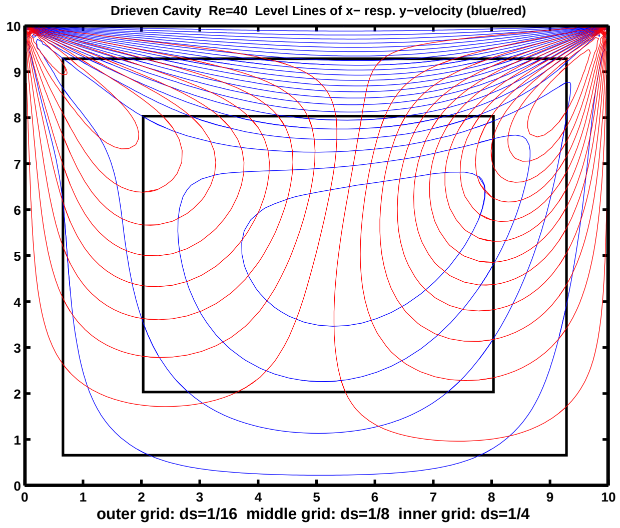


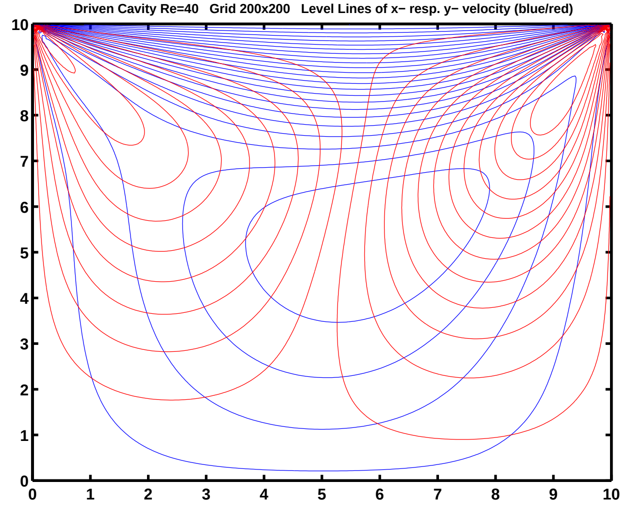


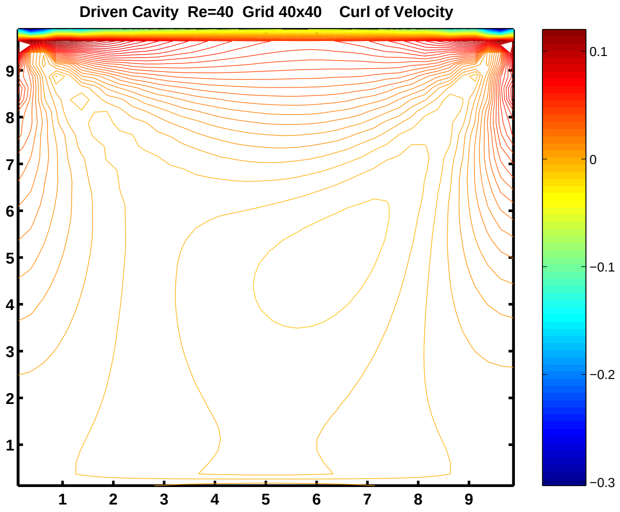


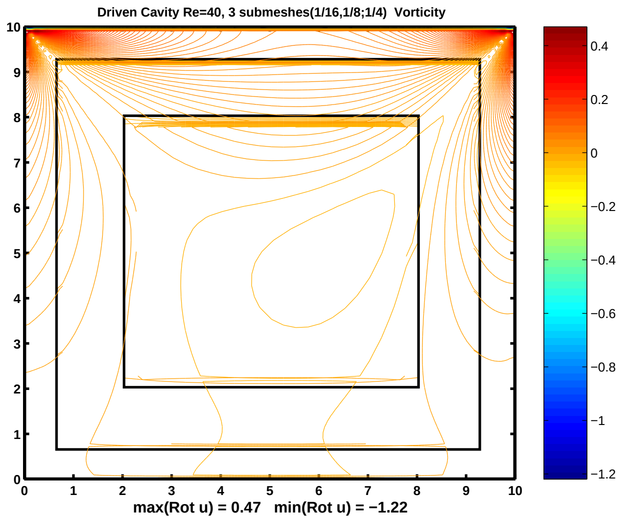
Lid Driven Cavity $Re = 40$, LBGK

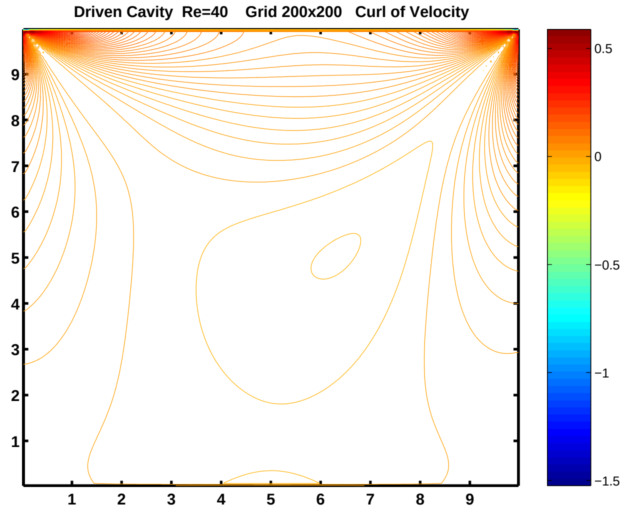




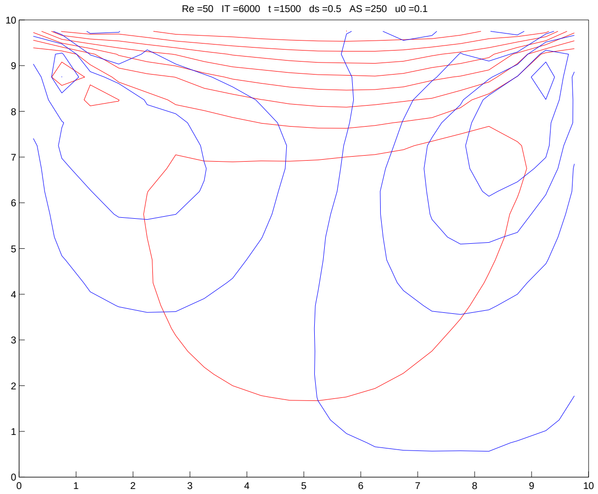


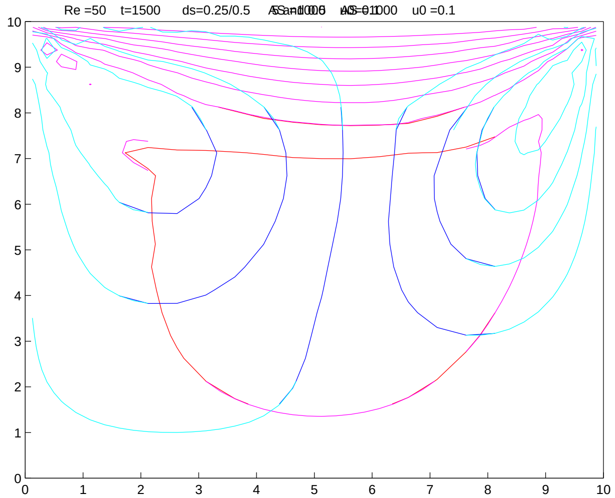


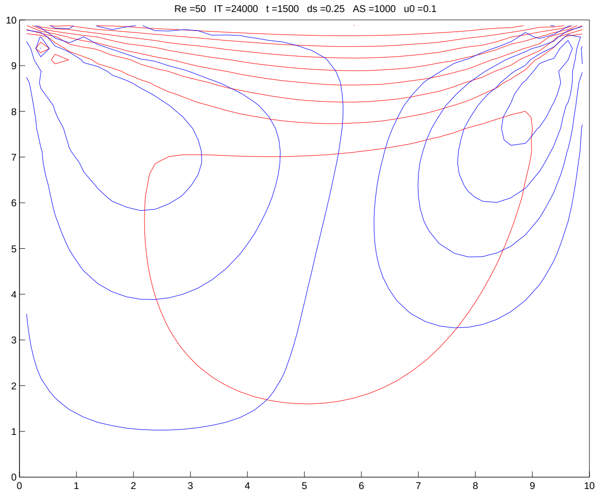




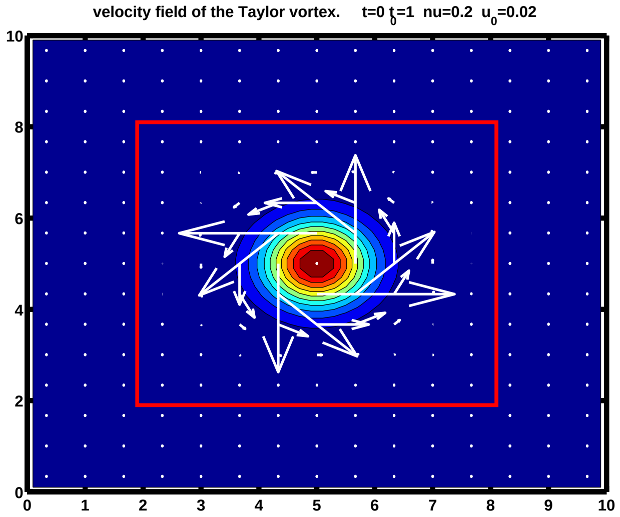
Lid Driven Cavity $Re = 50$, LBGK

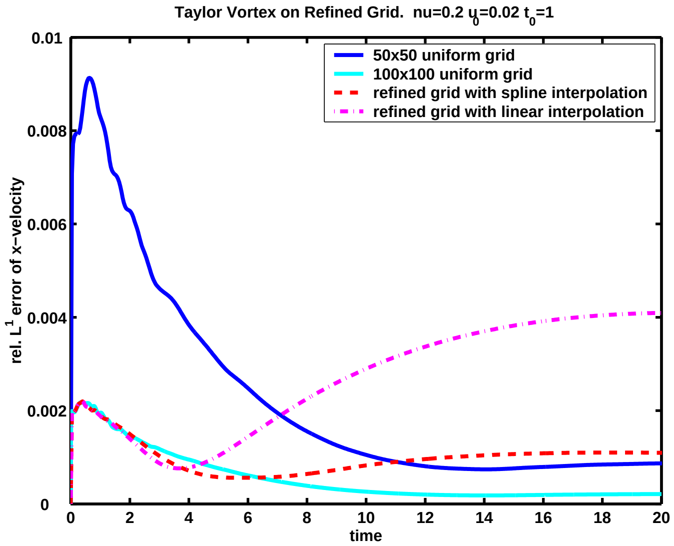


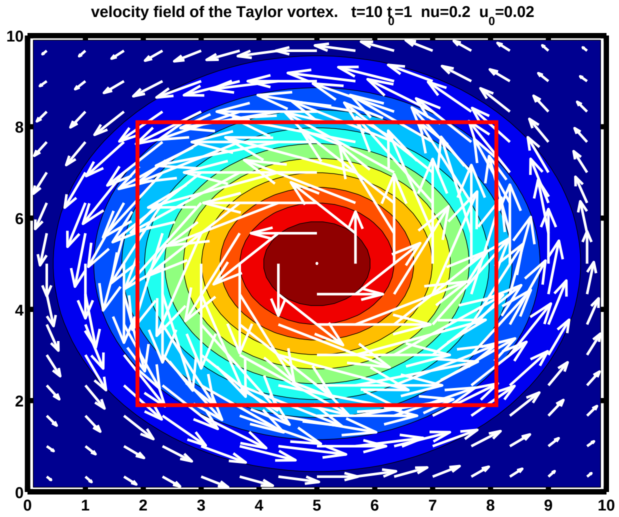




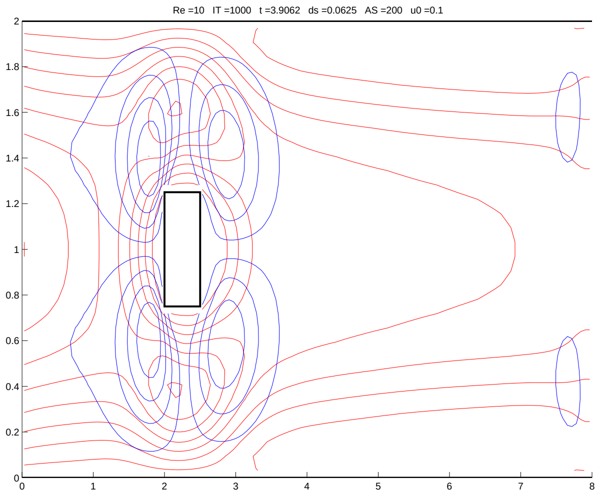
Taylor Vortex LBGK

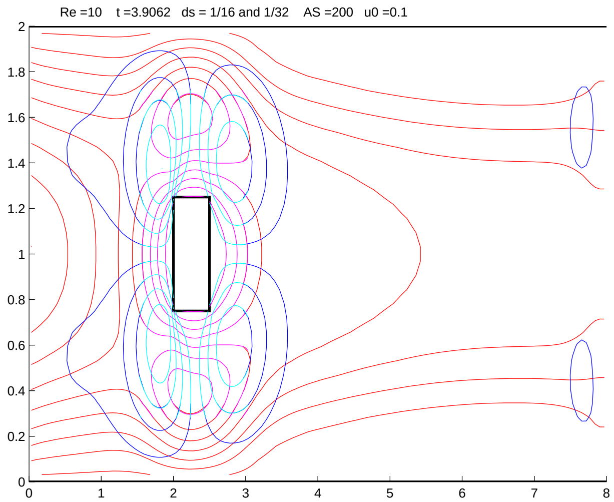


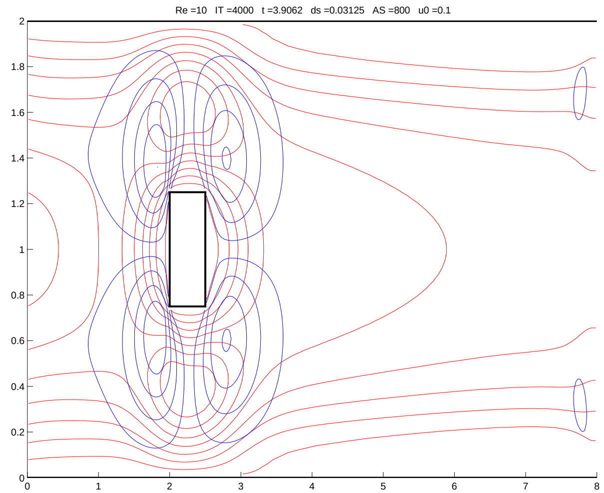




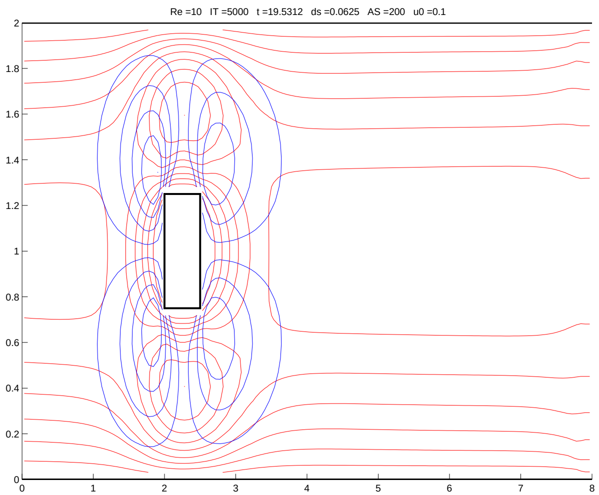
$Re = 10$, **LBGK**

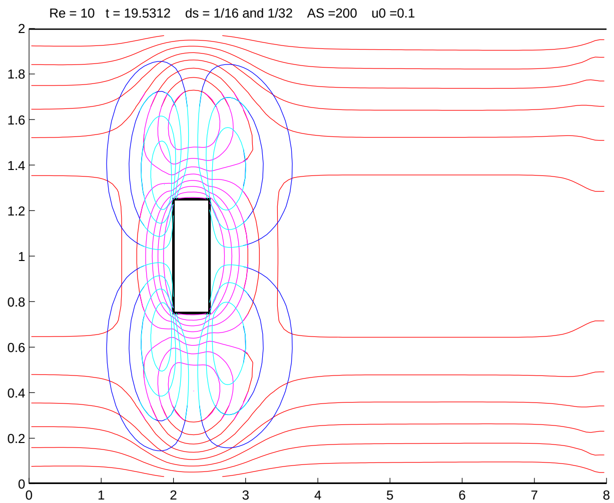


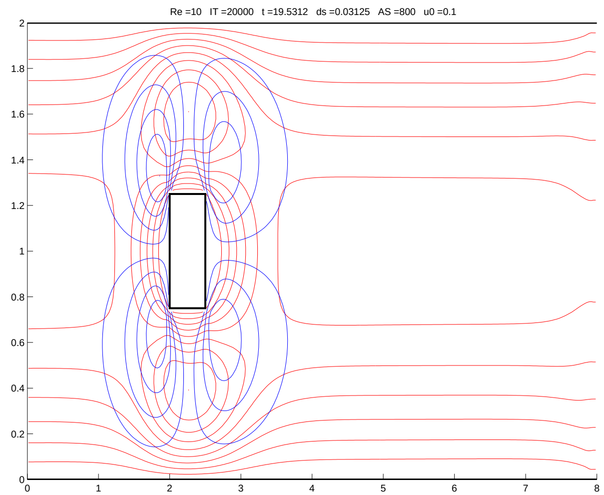




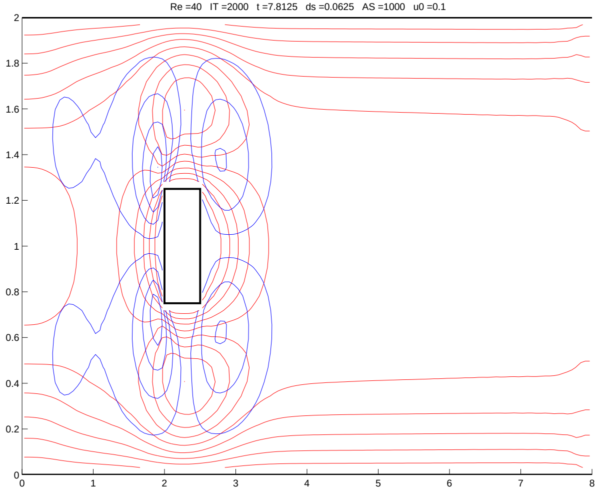
$Re =$ **LBGK**

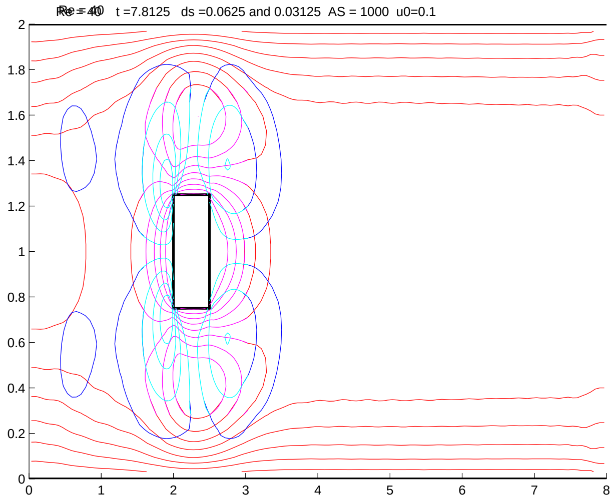


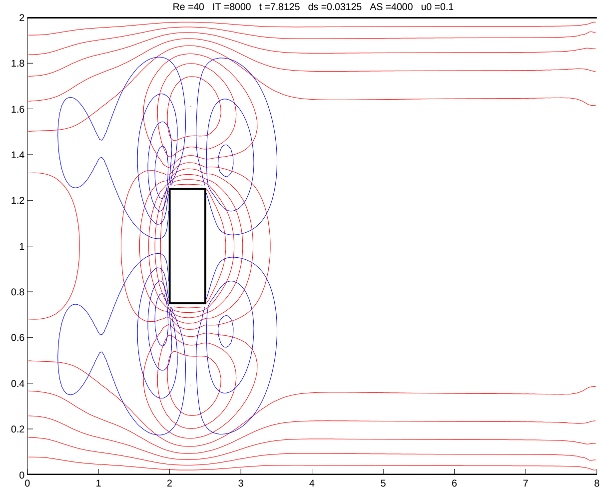




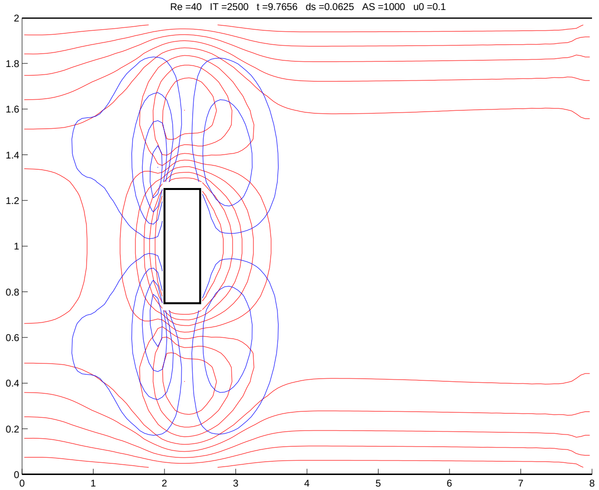
$Re = 40,$ **LBGK**

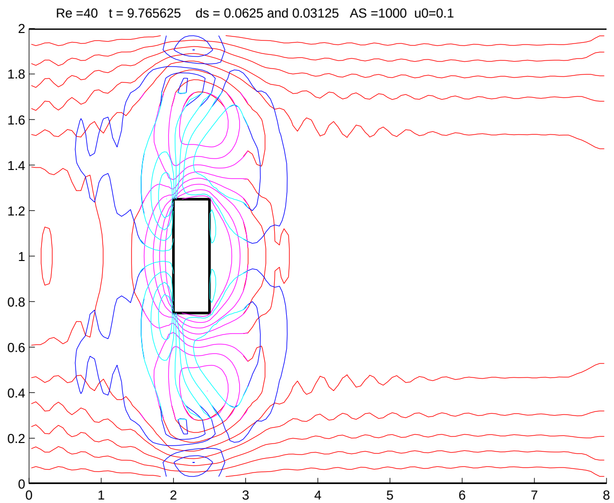


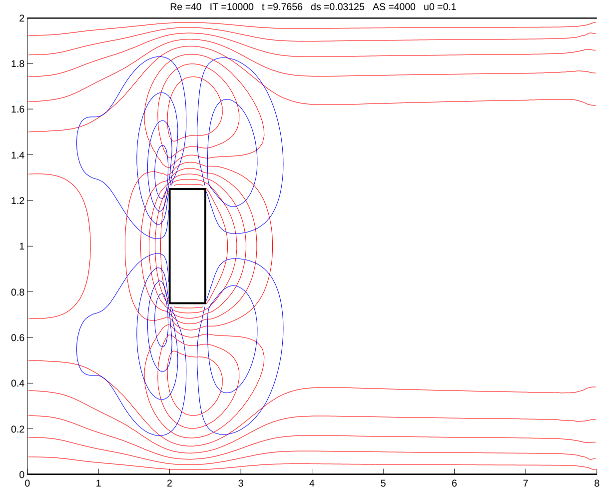




$Re = 40,$ **LBGK**







Student Version of MATLAB

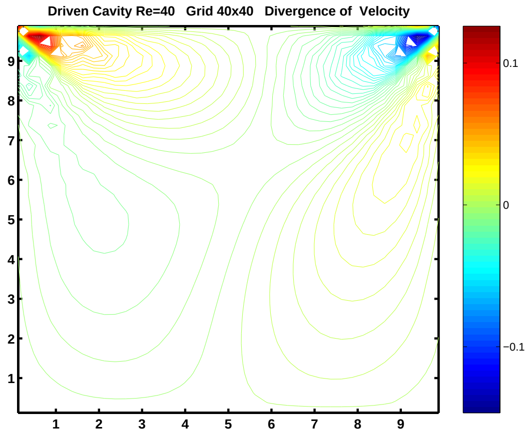
Lid Driven Cavity: Divergence of Velocity Field

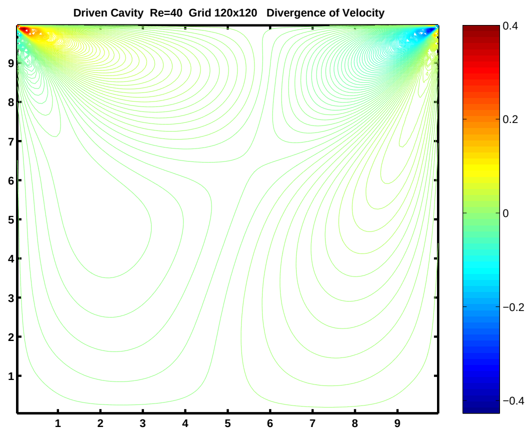
grid size	40×40	80×80	120×120	160×160	200×200
mean abs. value:	8.843e-3	9.347e-3	9.533e-3	9.631e-3	9.693e-3
maximum norm :	1.493e-1	2.906e-1	4.293e-1	5.675e-1	7.056e-1

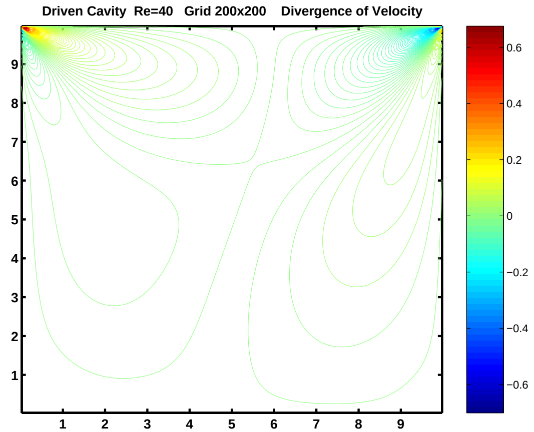
$$Re = 40$$

grid size	40×40	80×80	120×120	160×160	200×200
mean abs. value:	9.368e-3	9.808e-3	9.979e-3	1.007e-2	1.013e-2
maximum norm :	1.621e-1	3.123e-1	4.564e-1	5.983e-1	7.394e-1

$$Re = 100$$







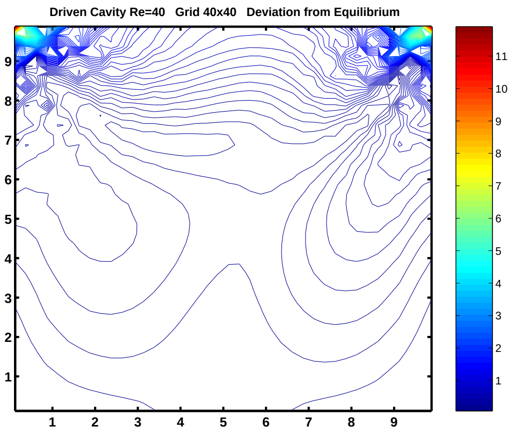
Lid Driven Cavity: Deviation from Equilibrium

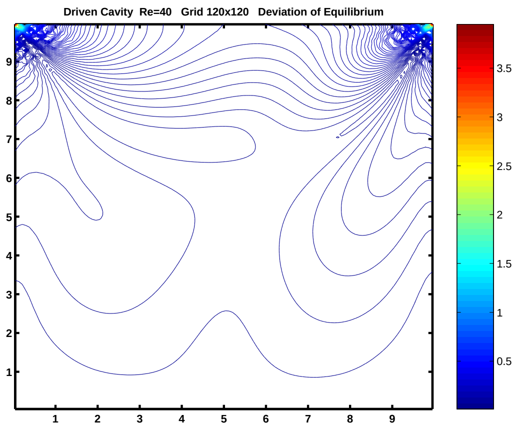
grid size	40×40	80×80	120×120	160×160	200×200
mean abs. value:	3.509e-4	8.554e-5	3.773e-5	2.115e-5	1.351e-5
maximum norm :	1.191e-2	5.932e-3	3.949e-3	2.960e.3	2.367e-3

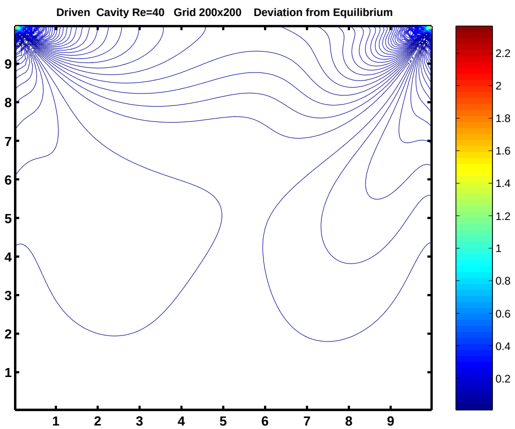
$$Re = 40$$

grid size	40×40	80×80	120×120	160×160	200×200
mean abs. value:	3.852e-4	8.785e-5	3.784e-5	2.096e-5	1.329e-5
maximum norm :	1.124e-2	5.606e-3	3.734e-3	2.799e-3	2.238e-3

$$Re = 100$$







Outlook