For office use only	Team Control Number	For office use only
T1 ________________	**43843**	F1 ________________
T2 ________________		F2 ________________
T3 ________________	Problem Chosen	F3 ________________
T4 ________________	**A**	F4 ________________

2016 MCM / ICM

Bathtub Simulation Using the LBGK and Visualization Method

Summary

Based on the Lattice Boltzmann BGK Method (LBGK) and a computational to find an optimal solution for heating a bathtub of cooled water.

First we build a velocity field models to simulate the convection of the water. By introducing the D2Q9 Lattice Boltzmann model, we obtain an applicable prototype depicts the streaming and collision of the particles in the water. We manages to apply a bounce back boundary conditions and visually depict the convection in the bathtub.

Then we combine the velocity model with a temperature model, which utilizes a D2Q9 scheme and the same boundary conditions as the velocity model. We couple the two model with the buoyancy term in the momentum equation, simulating the bathtub environment as we use hot water to heat the already cool water. The model is able to analyze the effects imposed by some conditions like the volume of the bathtub, the volume and the temperature of the person and the dissipation of the surface. It is worth noticed that by variating the viscosity coefficient in the coupled model, we manage to simulate the situation when bubble bath additives are added to the bathtub. Hence the model is flexible to test different conditions and figure out the best strategy.

In addition to the visualization in the velocity field, we build the visualization of the temperature field via Fluent®. Using the visualized model, we test the effects of the shape of the bathtub and the motion on our goal.

Finally, through the analysis comparison of different conditions, we come up with a combination of strategies to reach the target temperature and an evenly distributed temperature field with the least amount of water.

Guidance

To the User of the Bathtub

To whom it may concern,

What an exciting news to get a brand new bathtub in your home! Think about it, a nice and cozy hot bath at night and get rid of all the tiredness of the day.

Apart from all the happiness, you know that a comfortable hot bath always takes a large amount of water. I am here now telling you another good news: this is not entirely truth! By taking our bath strategy, you can not only have a nice bath, also save the water as much as possible.

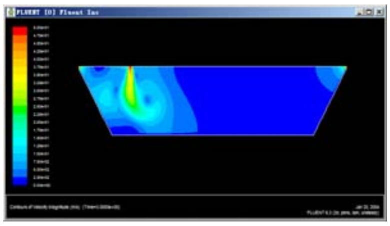
My workmates and I havve been studying deeply into the water convention and heat transfer of the water in the bathtub, as well as the shape of the bathtub itself, and the affection of bubble bath. A person's motion is also taken into consideration. Luckily, we finally found a win-win strategy containing what you need to do in the tub. We hope some of the outcome of our study can help you enjoy a hot bath, and at the same time, save the planet. This is the reason we writer this guidance.

Successfully, we make those difficult science into practical usage, and then we make it even more easily-understanding. A visualized simulation model means you can actually see what may happen in the bathtub if you follow our strategy. There are only two simple movement you need to do, open the faucet when you feel cold and move you hands or legs around when you get tired of lying still.

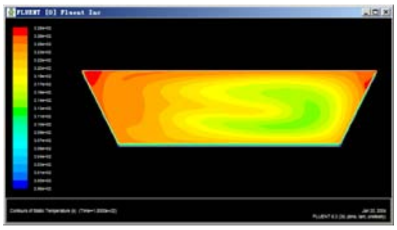
For the first movement, we highly suggest that you put a thermometer in the bathtub to take control over everything. The temperature you may feel cold is around 30°C. When you feel cold, or the thermometer drops to 30°C, all you need to do is to open the faucet and let the hot water in. The temperature of the hot water is around 70°C, which is also a part of our study. Afterwards, relax and enjoy you bath.

For the next movement, which is you motions, is relatively random. According to our study, it is much better to move left-and-right other than up-and-down to make the temperature of water even in a short time.
This is because the length of a bathtub is considered to be much greater than the width, so horizontal water needs to be stirred more. Just make a few movements when you gets bored.

To conclude, we hope by taking our bath strategy, you can have a nice hot bath without wasting too much water. We sincerely hope that with everyone's efforts, including our study and your wisdom of the new strategy, we can finally reach the aim of environmentally friendly lifestyle.



5s



500s

Contents

1 Introduction

1.1 Background

Since the porcelain was enamelled to cast-iron bathtub has been successfully marketed in 1880s, the bathtub is now widely used all over the world. According to the data in 2014, the total consumption of bathtub is over 9 billion square meter globally [Wikipedia]. In China, almost every family has at least one bathtub. With the development of human engineering in 20th century, a functionally designed bathtub has become a part of our life.

Although the spa-style bathtub with a heating system and circulating jets is on the market, the majority is still the normal ones. That means it is a simple water containment vessel with a water-inlet faucet and an overflow drain. When the water gets colder in the tub, one may open the faucet inlet the hot water. Similarly, excess water escapes though an overflow drain when the tub reaches its capacity.

So as to enjoy a comfortable bath without wasting water, some factors are utterly essential. The time to open the faucet (t), the temperature (T) and speed (V) of the inlet water, alongside the motions of the person should all be taken into consideration. Nevertheless, even though the awareness of water sustainability has been raised among people, not many would are how much water could be wasted when taking a bath. In this situation, the strategy including all the factors mentioned should be considered.

1.2 Previous Research

In the field of computational fluid dynamics(CMD) research, the conservative approach is the *Navier-Stokes Equations*. Integrated with the Newton second Law and a pressure term to describe the viscous flow, this equation provides a macro-level and continuous solution for the motion viscous fluid substances . However in this scenario where the shape of the container is uncertain, a macro model seems clumsy and redundant. Therefore it's expedient to apply a discrete and computer-friendly method — the Lattice Boltzmann Method to analyse the problem.

Unlike solving the Navier-Stokes equations, Lattice Boltzmann method (LBGK) is a numerical method that models the complex fluid flows. As a discrete simulation of a set of continuum equations, LBGK utilizes the evolution equations of propagation and collision for discrete time, resembling the Autonomous Cellular method.

The idea of this method originates from Boltzmann's gas and fluid theory. The convection of the gas and fluid is considered as the random motions of the particles. It simplifies the random motion by confining the particles into lattice and models the motion as propagation and collision. Additionally, the LBGK method towers many other computational fluid dynamics method because it can incorporate with flexible boundary conditions.

In single distribution model of LBGK, only mass and momentum conservations are considered, but the thermal effects are not negligible in many applications. Some researchers uses a MS model, i.e., the multiple velocities model to incorporate temperature into LBGK. [1] introduces additional discrete velocity and some higher order terms of velocity to obtain the energy equations, but it is restricted by the small temperature range and instability of numerical simulation results. In [2], the author employs a multiple distribution function model (MDF) by coupling a temperature distribution with the previous velocity distribution. Using this method, [3] successfully simulates the Rayleigh-Benard convection for a range of Reynolds numbers. The velocity distribution is this paper is modelled with 2 dimensions and 9 microscopic directions, known as a D2Q9 model, and the temperature distribution is with a D2Q4 model. This paper uses these two models to simulate the convection effects of the bathtub and computes the optimal strategy for the problem. In addition to the Lattice Boltzmann method, we apply CFD of Fluent to simulate the situations that the Lattice Boltzmann fails to cover, including the effect of the shape of bathtub and the stirring effects on the water.

2 Assumption and Justification

We make the following assumptions to make the problems simple and accessible.

1. **The effects of temperature on the density only affects the buoyancy term.** This is the commonly-used Boussinesq approximation [3].
2. **The bath tub is of simple geometry.** Bathtubs are usually cuboid with some curvature. The curvature of commercial bathtubs are often small for economic reasons.
3. **The density of water doesn't change much.** The density of water ranges from nearly 1 g/cm^3 to 0.995 g/cm^3. Hence we can safely neglect the change.
4. **The walls of bathtub has no dissipation effect.** Commercial bathtubs are designed to be heat insulated. The dissipation is negligible.
5. **Water in the bathtub is laminar flow.** The water in the bathtub is relatively slow, which enables us to neglect the existence of turbulence so that the computation can be greatly simplified.
6. **The surface of the water is flat.** Despite some small disturbance, the water is mostly calm in the bathtub.

3 The Prototype

3.1 The Velocity Model

3.1.1 Background and Principles of the prototype

In this section we are going to specify the *Lattice Boltzmann Method (LBM)*, which is closely related to the research on water circulation in the bathtub.

The general idea of the LBM is imaging the fluids is composed of a huge number of tiny fluid particles, each of which is moving in an arbitrary direction. LBM can be schemed to better describing the collision with irregular bounds. When colliding, we can acquire the exchange of momentum and energy by "particle streaming and billiard-like particle collision"[4]. The equation below characterized the exchange process.

$$\frac{\partial f}{\partial t} + \vec{u} \cdot \nabla f = \Omega$$

in which the $f(\vec{x}, t)$ denotes the particle distribution function , $\vec{u}$ the particle velocity and Ω the collision operator. The problem became discrete and therefore simpler by confining the moving particle in a lattice, and assume there are only limited possible moving direction(including staying still) on the next moment of the specifying particle.

To make the first step easier, we first constructed a *D2Q9* model, i.e. there are two dimensions and nine velocity factors. The velocities are defined as

$$\vec{e_i} = \begin{cases} (0,0) & i = 0 \\ (1,0),(0,1),(-1,0),(0,-1) & i = 1,2,3,4 \\ (1,1),(-1,1),(-1,-1),(1,-1) & i = 5,6,7,8 \end{cases}$$

In our Matlab simulation, we setted the direction of the vectors as the figure shows, plus $\vec{e_1} = 0$ which denotes the particle stay still.

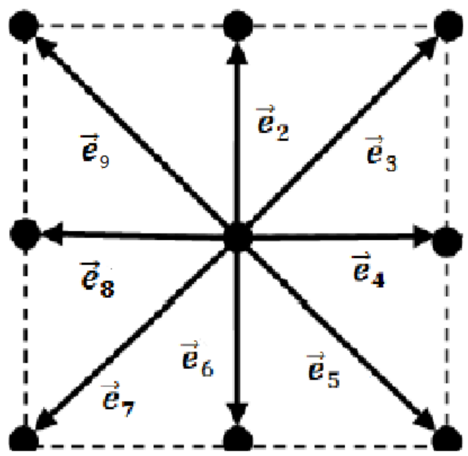


Figure 1: D2Q9 model illustration in our MATLAB code

Also several parameters are essential in our model:

$$\rho(\vec{x}, t) = \sum_{t=0}^{8} f_i(\vec{x}, t)$$

is the *microscopic fluid density* that characterizes the particle distribution. And

$$\vec{u} = (\vec{x}, t) = \frac{1}{\rho} \sum_{t=0}^{8} c f_i(\vec{x}, t) \vec{e_i}$$

is the average amount value of the microscopic velocities. This can be shown as vector in the Matlab simulations because it represent a general description of the particles within a lattice node.

In a D2Q9 Model, we model the collision process according to the equation below:

$$f_i(\vec{x} + c\vec{e_i}\Delta t, t + \Delta t) - f_i(\vec{x}, t) = -\frac{[f_i(\vec{x}, t) - f_i^{eq}(\vec{x}, t)]}{\tau}$$

where τ is the time to reach local equilibrium, and $f_i^{eq}(\vec{x}, t)$ the equilibrium distribution that can be derived from the *Bhatnagar Gross Krook* (BGK) collision

$$f_i^{eq}(\vec{x}, t) = \omega_i \rho + \rho s_i(\vec{u}(\vec{x}, t))$$

where $s_i(\vec{u})$ is defined as

$$s_i(\vec{u}) = \omega_i \left[2\frac{\vec{e_i} \cdot \vec{u}}{c} + \frac{9}{2}\frac{(\vec{e_i} \cdot \vec{u})^2}{c} - \frac{3}{2}\frac{(\vec{u} \cdot \vec{u})}{c^2} \right]$$

the ω_i is called weight and therefore defined as

$$\omega_i = \begin{cases} 4/9 & i = 0 \\ 1/9 & i = 1, 2, 3, 4 \\ 1/36 & i = 5, 6, 7, 8 \end{cases}$$

3.1.2 Prototype Construction

We tried to detect some optimized solutions when ambiguously changing several distinguished variables including the shape of the bathtub(2D boundary), the position of the faucet and the speed ,direction of the inter flow. We hold to generate some ideas for the method of designing bathtubs.

The Prototype examines the flows inside the bathtub during different time periods. We will see how the water flows forms the fluid convection.

To simplify the qualitative analysis, our group made several ideal assumptions

1. Refined the model into 2D dimension, i.e a D2Q9 model
2. Assume there are no outside forces or interruptions, including human rotation, bubbles or winds, accidental impact etc.
3. Assume zero gravity involved in the model (Later we will prove that the influence of the gravity acceleration is too faint to calculate). And therefore no buoyancy. Only force of collision included.
4. Neglect the influence of temperature

Based on the Lattice Boltzmann Model specified in the previous subsection, we are able to create an iteration process to follow the tracks of the flowing regulation. The algorithm lies below:

First we constructed a common bathtub in a graph of $56 * 16$ lattice. **The vector denotes the aggregated velocity $\vec{u}$ within certain lattice node.**

Algorithm 1 LBM simplified algorithm

Input: *NX, NY, rho, deltaX*

Initialize: *NX=56; NY=16; OMEGA=1.0; rho0=1.0; W=[4/9,1/9,1/36,1/9,1/36,1/9,1/36,1/9,1/36]; cx=[0,0,1,1, 1, 0,-1,-1,-1]; cy=[0,1,1,0,-1,-1,-1, 0, 1];* $deltaUX = 10^{-6}$*;; iterator = 0*

Output: *UX UY*

while iterator < specified iteration(50 or 4000 etc.) **do**

1. Streaming $f_i \to f_i^*$ in the direction of $\vec{e_i}$
2. Calculate microscopic fluid density rho and the x,y coordinate of the average microscopic value of the microscopic velocities $\vec{u}$.
3. Derive f_i^{eq} of the BGK collision and compute the alternated distribution of function

$$f_i = f_i^* - \frac{1}{\tau}(f_i^* - f_i^{eq})$$

end while

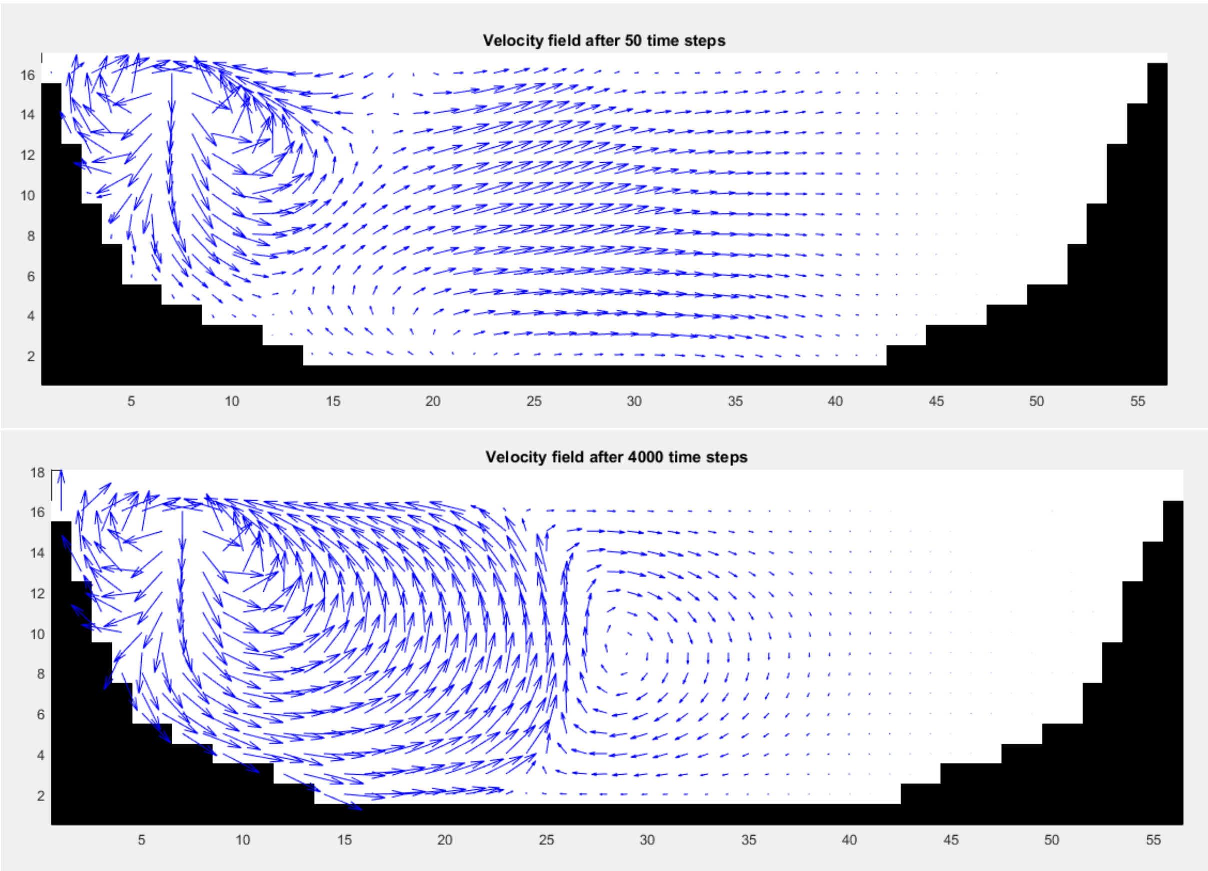


Figure 2: The flow of the common bathtub with side faucet

We first applied the notion of time step, because its a qualification analysis and later we can transfer time step into the real time. In the program, the while loop of iteration operates once as the time step increased by one.

First we set the faucet at one side of the bath tub, noticing that the fluid will goes into a steady state when we iterates the loop about 4000 times. Before the final steady state, we can observe that the swirl occurs near the arc boundary.We call this generating swirl. Near the generating swirl , there always gradually forms another circle that has an opposite rotation of the generating swirl. The anti-rotation circle is much weaker than the generating swirl. In our instance, the velocity of the anti-rotation swirl is about $1/100$ of the that of the generating swirl.Though our observation we found that the anti rotation circle is hard to formulate because of the distance of the base of the bathtub is so long ($15 \sim 43$ in x coordinate of this graph), so the flow just "propagates on the base of the bath tub" and hard to rotate.

To prove our assumption of the formation of the anti-circle, we find an even extreme instance – the rectangular bathtub (there are no arcs in a rectangular bath tub)

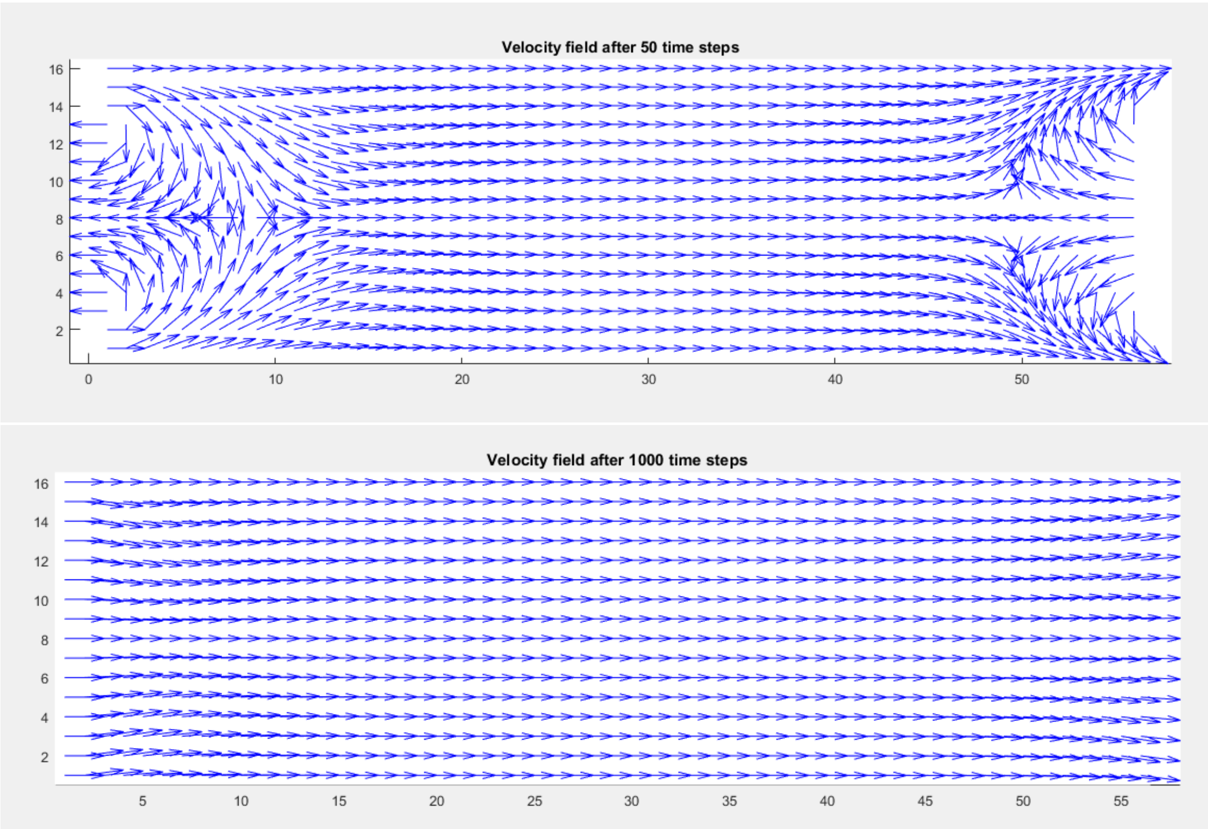


Figure 3: The flow of the common bathtub with side faucet(ejecting towards right)

As we can see in the extreme case, its not possible to form a circulation of flow after steady . Therefore it's wise to elaborately design the boundary curve of the bathtub.

There is one thing to mention, **there are two reasons** that we want to increase the velocity of the fluid. The first one is to try to acquire a even and smooth temperature throughout the bathtub. The second one is to create a feeling of motion when bathing, because it should be more comfortable. The bathing can be considerate a kind of fluid massage.

Then we tried an convex base bathtub.

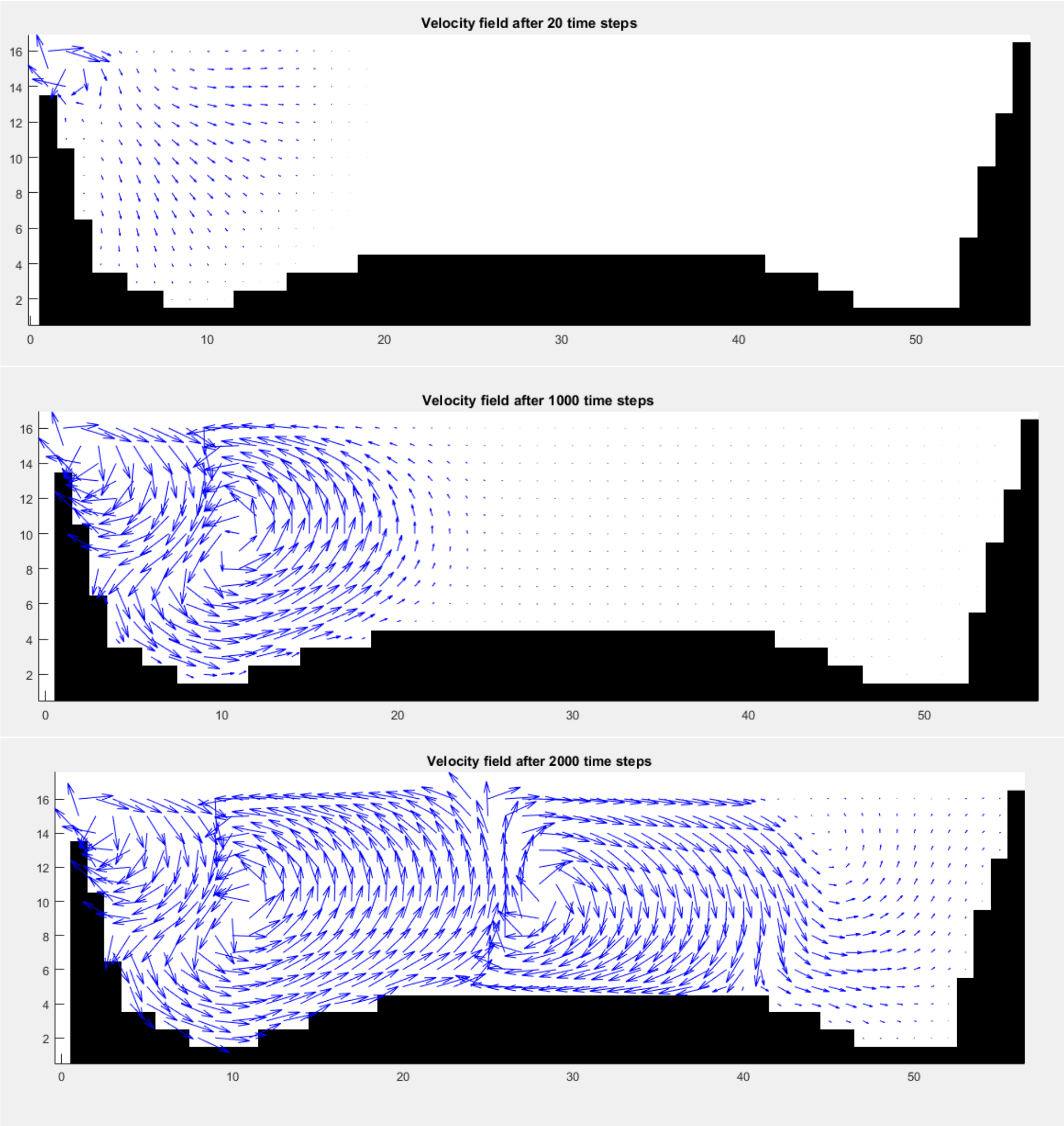


Figure 4: The flow of the convex bathtub with side faucet(ejecting towards right)

In the convex-base bathtub, we can clearly identify the three cycles when the time steps became 2000, which is a steady state. Each circulation in between are of anti-direction of rotation. Note that the threshold of the vector magnitude is modified in the "2000 time step" figure. The threshold is a critical amount, when the magnitude of a certain vector exceed this amount, the magnitude of this vector is shirked to be equal to this amount, whereas the vector whose magnitude is smaller than the threshold remains the same. To do this we can detect the relative size of the vectors that are too tiny to detect, while the large vectors are restricted so they can't be too large in the graph.

The last detection is that the perimeter of the circulation become small near the sharp angles. This can be shown in a "extreme base model" This can be shown in the figure. Also by producing the smaller cycles, we acquire more cycles within the "bathtub" of same size. Although apparently this kind of "bathtub" doesn't exist, this can really shown as a illumination of the further works.

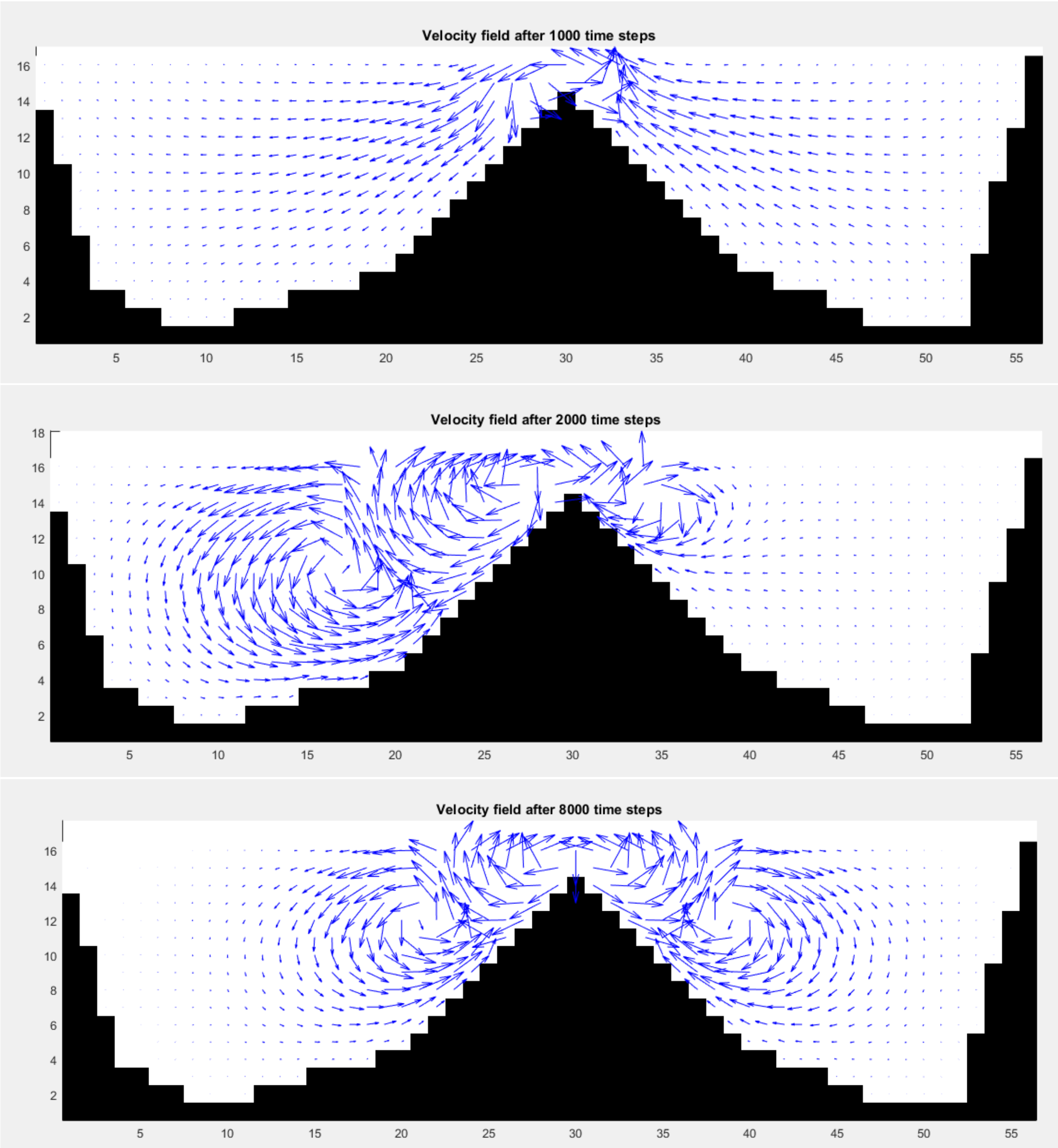


Figure 5: The flow of the extreme-convex bathtub with middle faucet(ejecting towards down)

3.2 The Temperature Model

3.2.1 The D2Q4 Model

In a similar manner to the velocity field, the temperature field can be modelled with BGK. If the viscous heat dissipation and compression work can be negligible, the following equation is followed [2]

$$\frac{\partial T}{\partial t} + \nabla \cdot (\mathbf{u}T) = \mathscr{D}\nabla^2 T \tag{1}$$

where $\mathscr{D}$ is the diffusivity constant, **u** is the velocity of the fluid and T is the temperature.

To interpret the equation, we build a D2Q4 BGK model, i.e., the each lattice has four distribution function $\mathbf{e}_1$,$\mathbf{e}_2$,$\mathbf{e}_3$ and $\mathbf{e}_4$. They point in four directions and indicate the propagation of the temperature. The propagation evolution equation is

$$T_i(\mathbf{x} + c\mathbf{e}_i\Delta t, t + \Delta t) = T_i(\mathbf{x}, t)$$

The evolution equation for particle collision is given by

$$T_i(\mathbf{x} + c\mathbf{e}_i\Delta t, t + \Delta t) = T_i(\mathbf{x}, t) - \frac{1}{\tau}(T_i(\mathbf{x}, t) - T_i^{eq}(\mathbf{x}, t)) \tag{2}$$

where τ is the relaxation time of the particle collision and c is the lattice velocity. The equilibrium temperature is then

$$T_i^{eq} = \frac{T}{4}(1 + 2\frac{T_i^{eq} \cdot \mathbf{u}}{c}) \tag{3}$$

The sum of the temperature distribution is then the temperature for the lattice, i.e.

$$T = \sum_{i=1}^{4} T_i \tag{4}$$

3.2.2 Coupling of the Velocity and Temperature Model

According to Boussinesq approximation, density are only effective for buoyancy term in the momentum equation, i.e.

$$\frac{D\mathbf{u}}{D\,t} = -\frac{1}{\rho_0}\nabla p + \nu\nabla^2\mathbf{u} + \frac{\rho}{\rho_0}\mathbf{g} \tag{5}$$

where ν is the kinetic viscosity and ρ_0 is the constant ambient fluid density. We can note that ρ only contributes to the buoyancy term, and

$$\rho = \rho_0(1 - \beta(T - T_0)) \tag{6}$$

where β is the coefficient of thermal expansion. Hence we can modify the velocity model by adding a buoyancy term cause by the temperature.

$$f_i(\mathbf{x} + c\mathbf{e}_i\Delta t, t + \Delta t) = f_i(\mathbf{x}, t) - \frac{1}{\tau}(f_i(\mathbf{x}, t) - f_i^{eq}(\mathbf{x}, t)) + b_i \tag{7}$$

where

$$b_i = -\frac{1}{2c}\Delta t\alpha \cdot \mathbf{e}_i \cdot \mathbf{g}\beta(T - T_0) \tag{8}$$

where $\alpha = \delta_{i2} + \delta_{i4}$. $\delta_i j$ is the Kronecker delta. To simplify, we will add the buoyancy term to the velocity distribution function with the up direction and subtract it from the down direction.

To determine the coefficient of the buoyancy term, we use the data of thermal expansion of water from [Wikipedia]. In our model, we set the lattice velocity to be 0.33 and the time interval to be 1s. Hence the coefficient is

$$b = \alpha \cdot g \cdot \beta \cdot \frac{1}{2c} = 95$$

3.2.3 Boundary Conditions

In this model we simply assume that the temperature field boundary condition is the same as the velocity field, that is, the bounce back boundary conditions. It is sensible to assume so because the temperature mainly propagates with the motion of particles.

4 Model Simulation and Computer Simulation

Based on Lattice Boltzmann method, we implement our model using Matlab. We start from a simple model. The height and length are set to be 10 lattice. The collision operator Ω is set as 1.0 to indicates the relaxation time is 1 time unit. The lattice velocity c is set as $\frac{1}{3}$. At first, the temperature of the water in the bathtub is 300K, and our target temperature is 330K. We add water from a point in the upper left with a temperature higher than the target temperature. The model keeps going until the fluid settles down, the temperature evenly distributes and the gap between the temperature and ideal one being in an acceptable range.

Later, we change the model to simulate more complex situations. Our model is so flexible that with some parameters alternative, it is applicable to other situations.

The effect of bubble bath additive is that it alters the viscosity coefficient of the water. Since in our model,

$$\nu = \frac{2\tau - 1}{6} \frac{(\Delta x)^2}{\Delta t} \tag{9}$$

Hence the relaxation time changes as the density of the bubble bath additive changes. In this way, we can simulate the effect imposed by the bubble bath additive.

For dissipation, we simulate it by forcing the water on the surface to go through a temperature decrease at a constant rate. Since the material of the bathtub is thermal conservative, we assume the bathtub walls do not cause dissipation of the heat.

To simulate the person inside the bathtub, we introduce additional block of bounds but let it keeps a constant temperature to simulate the body temperature. The water will bounce back when it hits on the person's body like normal bounds but the thermal exchanges remain unaffected. It hence successfully simulates the effect of the human body on the bathtub water.

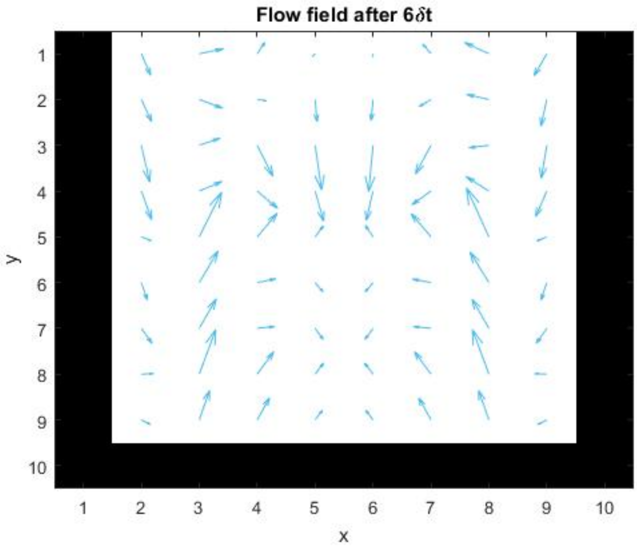


Figure 6: Simulation of the simple model

5 Optimal Solution Method Using Simulated Anneal Arithmetic

After constructing the integrated model of temperature and aggregated velocity in the bathtub, we need to find the optimized temperature when the fluid system become steady. Also we need to find the optimized solution for the least time to become steady state.

It's intuitively to construct two functions with two independent variables and one dependent variables. The two independent variables are the initial temperature of the inlet fluid and the initial velocity of the fluid.

An obstacle occurs immediately. The function that we will construct is so complexed that it can not be visualized by graphics. Therefore the traditional derivative method in high school Mathematics cannot apply. Alternatively, we would like to apply some probabilistic methods. And the most matured and exact method is the ***Simulated Annealing*** Method.

This method provide a high probability to find the optimized value. "It works by emulating the physical process whereby a solid is slowly cooled so that when eventually its structure is frozen, this happens at a minimum energy configuration"[5]

We assume the most comfortable temperature for human is 330K (This value can be adjusted by personal preference). Therefore the range of the initial inlet water temperature is from 330K to 360K. the magnitude of the aggregated velocity is less than $0.1m/s$ usually. We would like to find the minimum of the dependent values in this two-variable function. The algorithm of the simulated annealing in finding the minimum is as follows.

6 Results and Data Analysis

To obtain an optimal water control strategy for the bathtub problem, we control the parameters of inlet water velocity and temperature, to study their influence on time it takes for the temper-

Algorithm 2 Simulated annealing for minimum (partly cited[5])

Input: Inlet temperature T, the aggregated inlet velocity$|\vec{u}|$
Initialize: An infinite set of pairs $S = (T, |\vec{u}|)$,$T(t)$ the cooling temperature A real value costed function J defined on S
Output: The global minima set $S^* \subset S$

1. For each $i \in S$, a set $S(i) \subset S - \{i\}$ is called the neighbour of set S .
2. For every i , a collection of positive collection q_{ij}, $j \in S_i$ such that $\sum_{j \in S_{S_{(i)}}} q_{ij} = 1$. It's assumed that $j \in S(i)$ is only if $i \in S(j)$

while iterator < specified iteration(50 or 4000 etc.) **do**

1. if $J(j) \leq J(i)$, then $x(t+1) = j$
2. if $J(j) > J(i)$, then $x(t+1) = J$ with probability $e^{\frac{-J(j)-J(i)}{T(t)}}$ and $x(t+1) = i$ otherwise.
3. Formally we have

$$P[x(t+1) = j | x(t) = i] = q_{ij} e^{-\frac{1}{T(t)} max\{0, J(j)-J(i)\}}$$

if $j \neq i$ and $J \in S(i)$ **end while**

ature to be evenly distributed and be as close as possible from the cooled temperature (300K) to the target temperature (330K)in the bathtub and the temperature at the final stage.

6.1 Model without Heat Dissipation

6.1.1 Inlet Water Temperature

In the figure, as the temperature increases, the time it takes to reach the ideal state decreases. It is reasonable since higher temperature brings more heat to the cooled water in the bathtub and hence increases the temperature with a faster pace.

The final temperature of the bathtub water has a linear relationship with the temperature of the bathtub inlet water when the temperature is lower, as shown in the first part of the line in the figure. Since we cut off the inlet water when the ideal state is reached (within 3K), the later part all settles in the regime of 327K to 330K.

6.1.2 Inlet Water Velocity

The equilibrium time increases as the inlet water velocity increases. It is the case because it takes longer for the water to settle down in the bathtub if the water is more turbulent. The simulation favors a velocity of 0.01-0.02 L/s, as the figure indicates.

Surprisingly, the temperature decreases when the inlet water velocity increases. It can be ac-

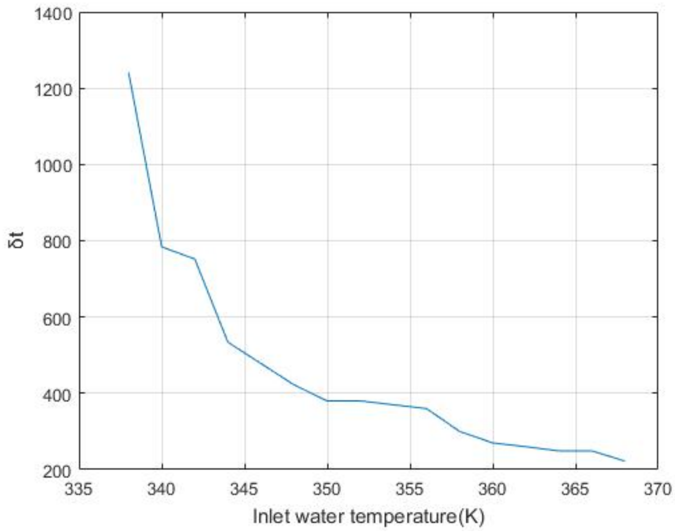


Figure 7: The relationship between the inlet water temperature and the time to reach an evenly distributed state. The inlet water velocity is controlled as 0.02 L/s

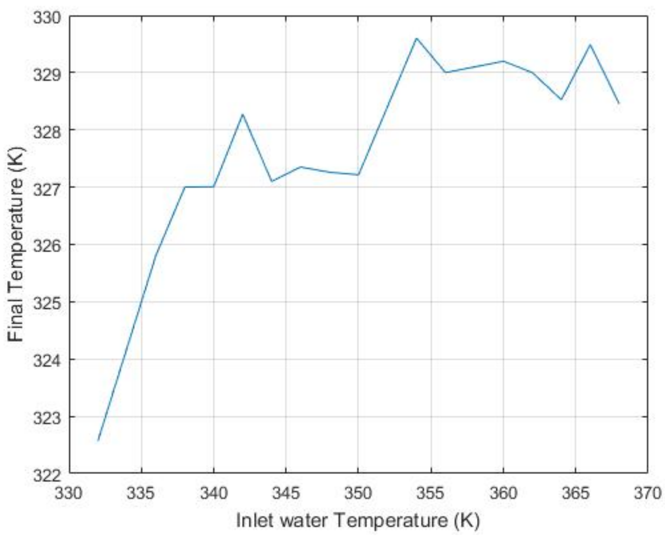


Figure 8: The relationship between the inlet water temperature and the final temperature in the bathtub. The inlet water velocity is controlled as 0.02 L/s

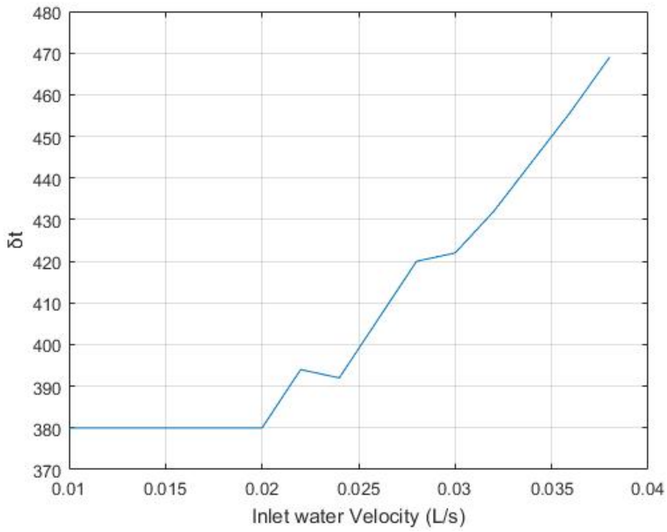


Figure 9: The relationship between the inlet water velocity and the time to reach an evenly distributed state. The inlet water temperature is controlled as 350K

counted by the fact that the larger water velocity tends to mix the whole tub of water together and results in a more evenly distributed, but lower temperature.

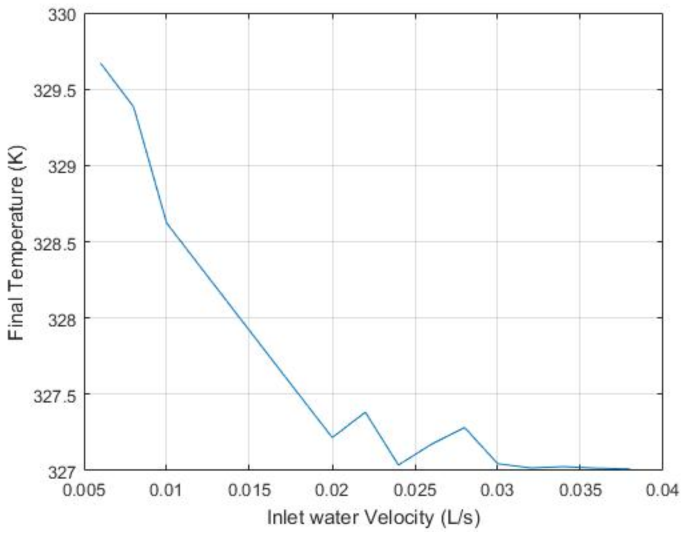


Figure 10: The relationship between the inlet water velocity and the final temperature. The inlet water temperature is controlled as 350K

6.1.3 Optimal Combination

Since we need to control the degree of the uniformity and the final temperature at the same time, single variable analysis can only provides us with a trend but not an optimal solution. Hence we employ the Simulate Anneal Arithmetic to obtain an optimal solution at (350K, 0.02L/s), i.e., when the inlet water temperature is at 350K and the velocity is 0.02L/s. This result coincides with the qualitative results we obtain in the previous part.

6.2 Model with Dissipation

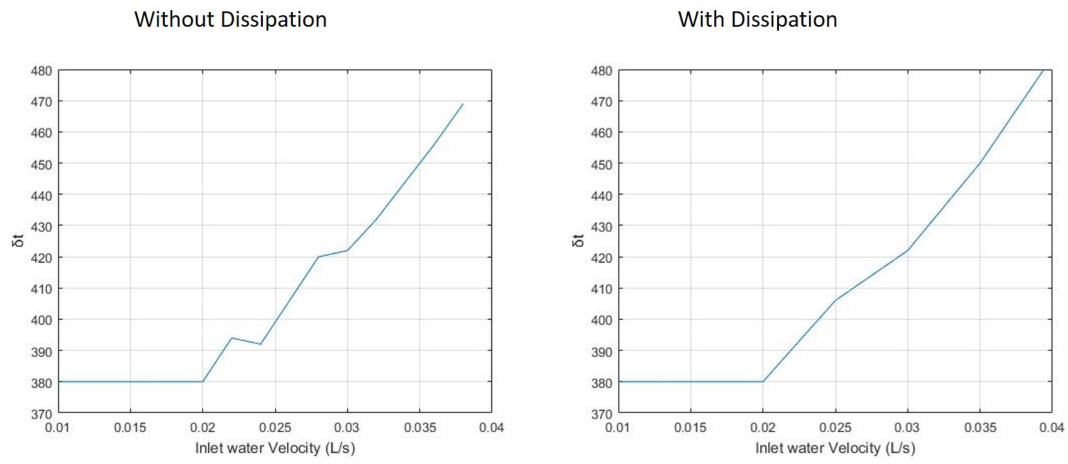


Figure 11: The comparison of models without and with dissipation. The inlet water temperature is controlled as 350K

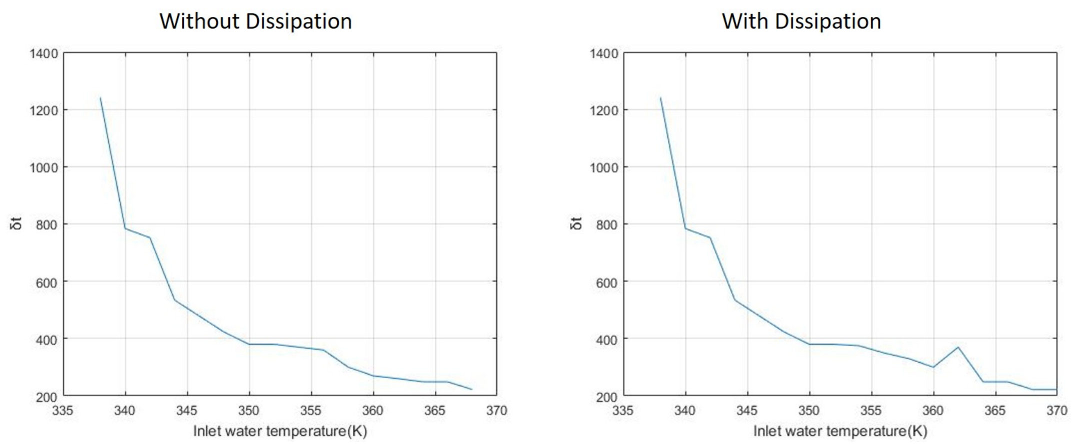


Figure 12: The comparison of models without and with dissipation. The inlet water velocity is controlled as 0.02L/s

As observed in the figures, the impose of dissipation does not affect the procedures too much. Dissipation is negligible compared to the effect caused by water we add. Hence from now on we only consider the model without dissipation, which is sufficient for application.

6.3 Effect of the Bubble Bath

Adding bubble bath additive can increase the viscosity of water and hence result in a higher relaxation time for the collision in the model. As shown in the figure, the bubble bath additive has no effect on the final temperature but has a relation with the equilibrium time. The equilibrium time decreases as the relaxation time increases. Hence it takes shorter if the density of the solution is higher. It is acceptable since the increasing viscosity confines the streaming hence the water velocity.

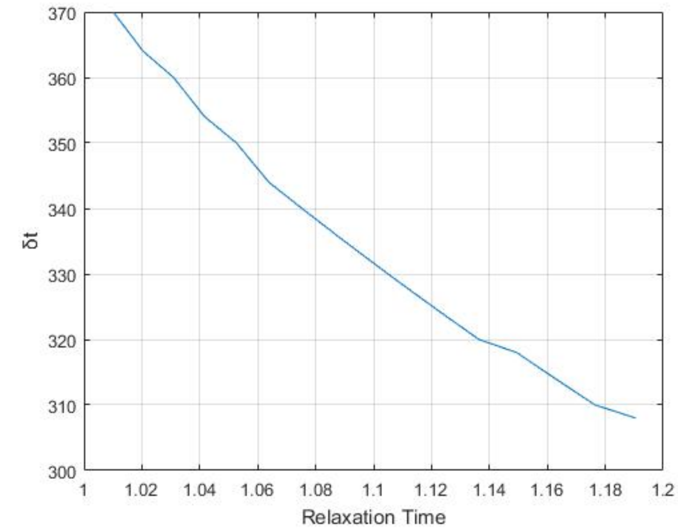


Figure 13: The relation between relaxation time and equilibrium time. The inlet water velocity is controlled as 0.02L/s, and the inlet temperature is controlled as 350K

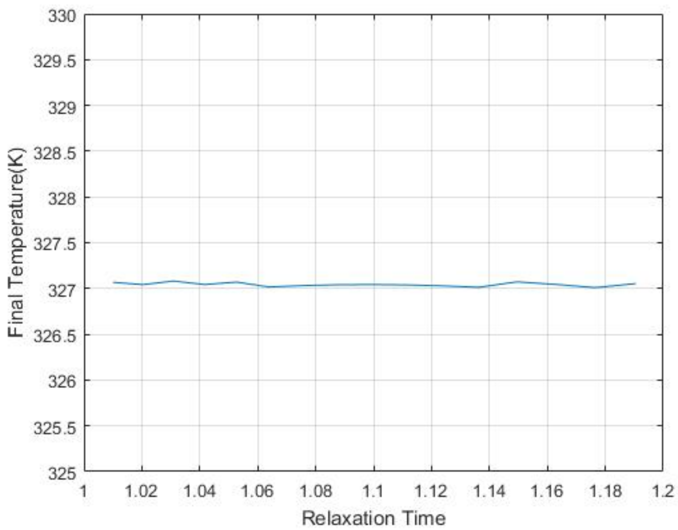


Figure 14: The relation between relaxation time and final temperature. The inlet water velocity is controlled as 0.02L/s, and the inlet temperature is controlled as 350K

6.4 Effect of the Bathtub Volume

The equilibrium time increases as the volume increases. Larger volume results in more water, hence it takes more time to heat the water. The final temperature is not affected by the volume.

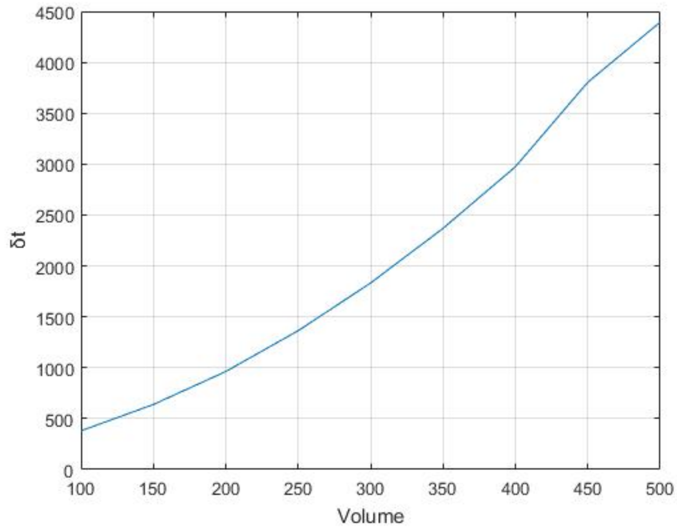


Figure 15: The relation between volume and equilibrium time. The inlet water velocity is controlled as 0.02L/s, and the inlet temperature is controlled as 350K

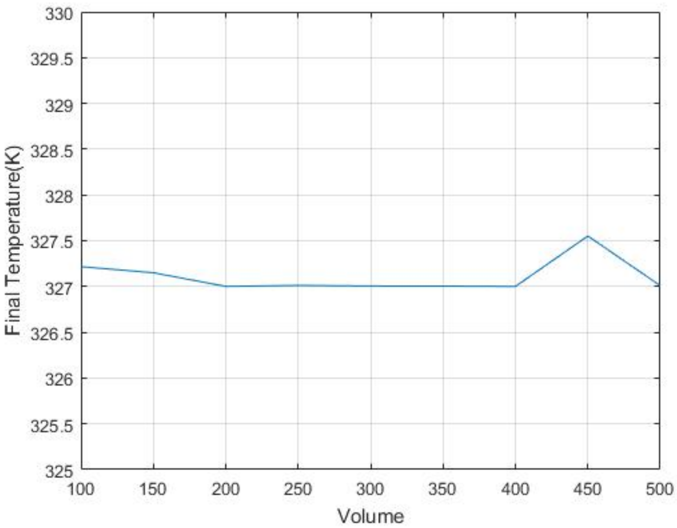


Figure 16: The relation between volume and final temperature. The inlet water velocity is controlled as 0.02L/s, and the inlet temperature is controlled as 350K

6.5 The Effect of the Person

We actually abstract the person as a cylinder and study him as a block, ignoring the actual size of the person. As human temperature increases the heating time decreases since the body can help heat the water, but there is no account for the fact that more time is required when the volume increases.

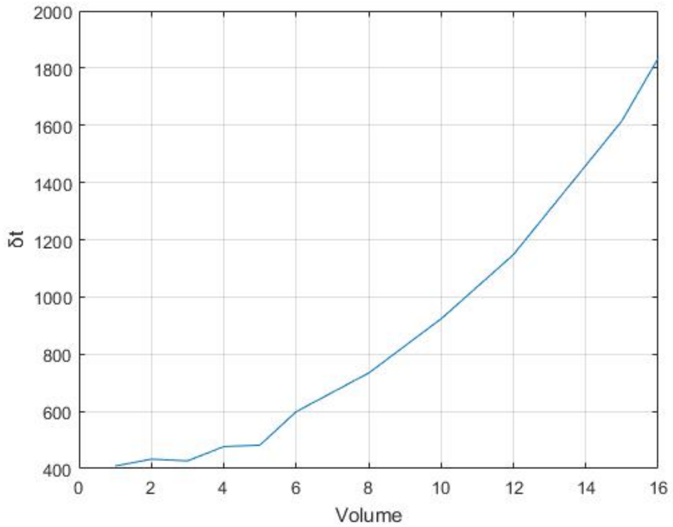


Figure 17: The relation between "human" volume and the equilibrium time. The body temperature is set to 310K

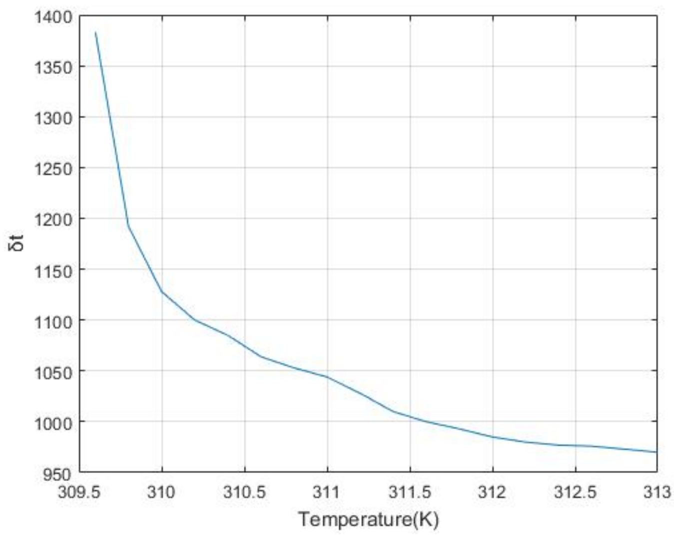


Figure 18: The relation between human body temperature and the equilibrium time. The volume is set to 10

7 Visualization of the Temperature Field

Generally, we build the model based on flow and energy equations. The equation for conservation of mass or continuity equation, can be written as follows[9]:

$$\frac{\partial \rho}{\partial t} + \nabla \cdot (\rho \mathbf{v}) = S_m \tag{10}$$

Where source S_m is the mass added to the continuous phase from the dispersed second phase.

Conservation of momentum in an inertial reference frame is written as

$$\frac{\partial}{\partial}(\rho \mathbf{v}) + \nabla \cdot (\rho \mathbf{v} \cdot \mathbf{v}) = -\nabla p + \nabla \cdot (\tau) + \rho \mathbf{g} + \mathbf{F} \tag{11}$$

where

$$\tau = \mu \left[(\nabla \mathbf{v} + \nabla \mathbf{v}^T) - \frac{2}{3} \nabla \cdot \mathbf{v} I \right]$$

Furthermore, FLUENT solves the energy equation in the following form:

$$\frac{\partial}{\partial t}(\rho E) + \nabla \cdot (v(\rho E + p)) = \nabla \cdot (k_{eff} \nabla T - \sum_j h_j J_j + (\tau \cdot v)) + S_h \tag{12}$$

In addition to equations, there are some more details related to the model. The pressure-velocity coupling is based on SIMPLE algorithm. For under-relaxation factors, the pressure is 0.3, density is 1, body forces is 1 and momentum is 0.7. For discretization, the pressure is standard, the momentum is first order upwind and the energy is first order upwind.

Additionally, the boundary conditions of the model is as follows.

Table 1: Boundary conditions

Zone	Type	Temperature	Velocity
water-inlet	velocity-inlet	350K	$0.5m/s$
water-outlet	pressure-outlet	300K	
water-surface	wall	298.15K	
tub-wall	wall	298.15K	
arm	wall	298.15K	$0.1m/s$

7.1 Shape Design of the Bathtub

Considering the visualized simulation of a 3D model costs much time and part of it is unnecessary given the symmetry of the bathtub, we compared three simplified 2D bathtub model in order to confirm which shape meets our expectation best.

The first is the rectangle tub in a 2D view. We use this type to illustrate how Finite Volume Method is used to create a visualized simulation. The next step is to build the mesh on the

Figure 19: The basic shape of the bathtub

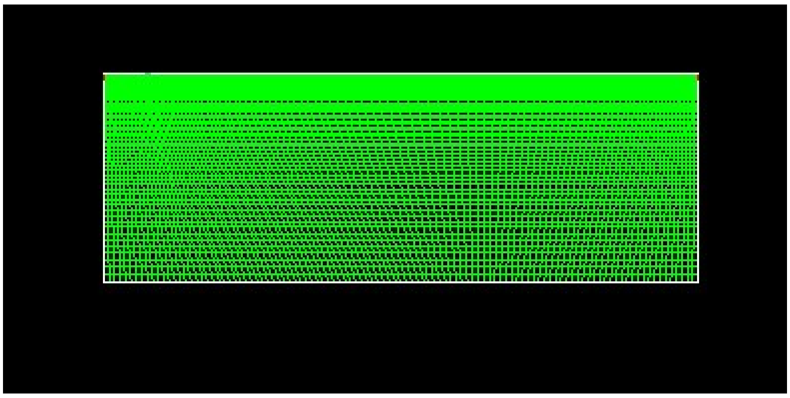


Figure 20: The grid

rectangle model, which concerns the computing results. First, every eges should be built on grid nodes so that the radio and intersize spacing should be carefully chosen.

After we built the mesh we need, it is time to consider all the boundary conditions and material. In order to control variable, we choose the temperature of inlet-water is 350K and the cold one Is 300K. The speed of inlet-water is 0.02L/s. The heat convention between the water and tub wall is taken into consideration. In addition, the model is defined unsteady. By adding the energy equation, we can obtain the visualized simulation of temperature in space and time. We chose 1s,3s,5s,10s,20s,35s to compare the even time.

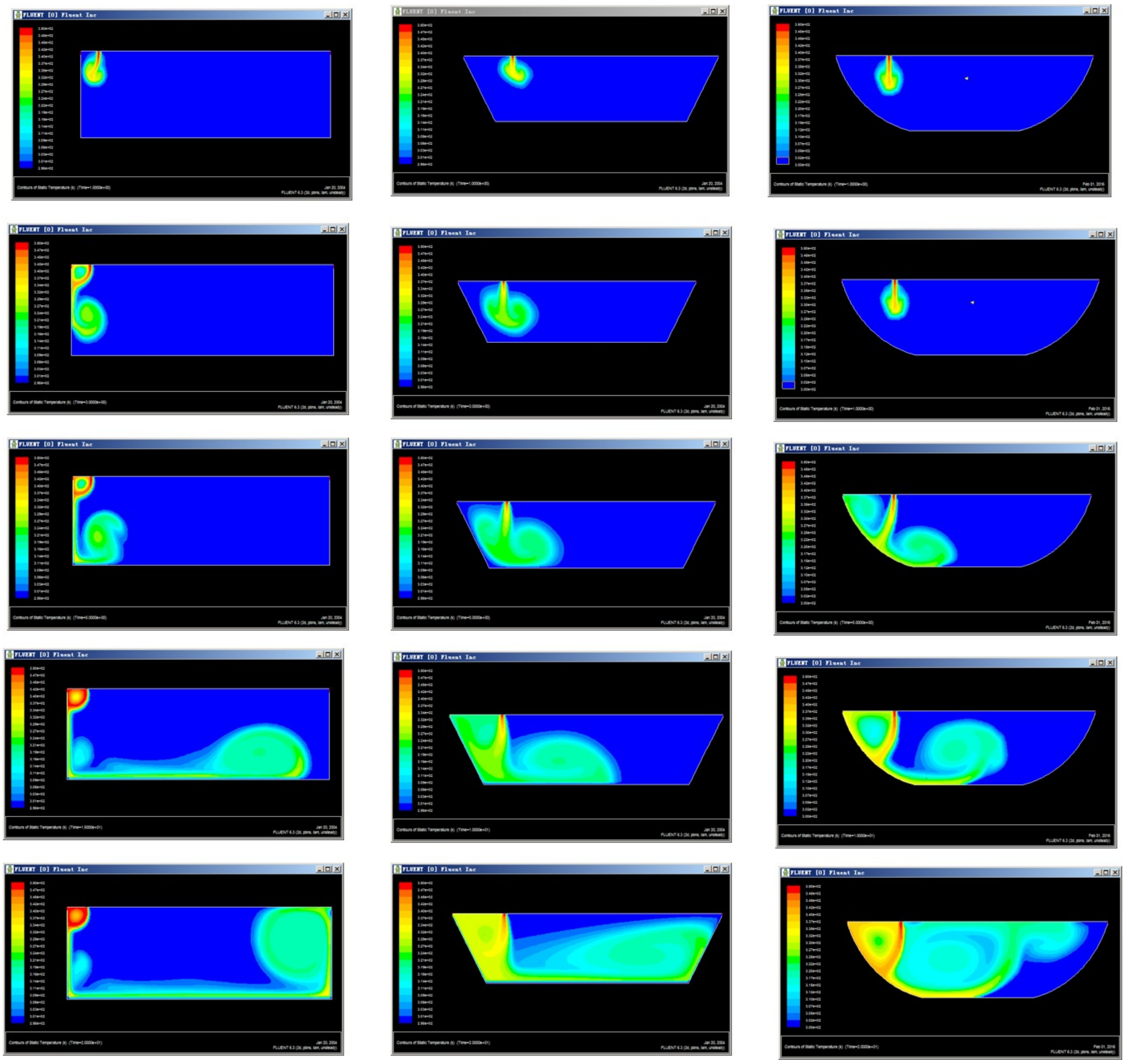


Figure 21: The second type bathtub is able to reach the initial water temperature, 310K, in the shortest time. It is difficult for the third bathtub to get an even temperature field, and the first one spend a rather long time to meet the goal. Hence the shape of the second bathtub is the best choice.

In addition, a 3D visualized simulation can be built with effort. We can see that the model is generally symmetrical so that a 2D model is qualified.

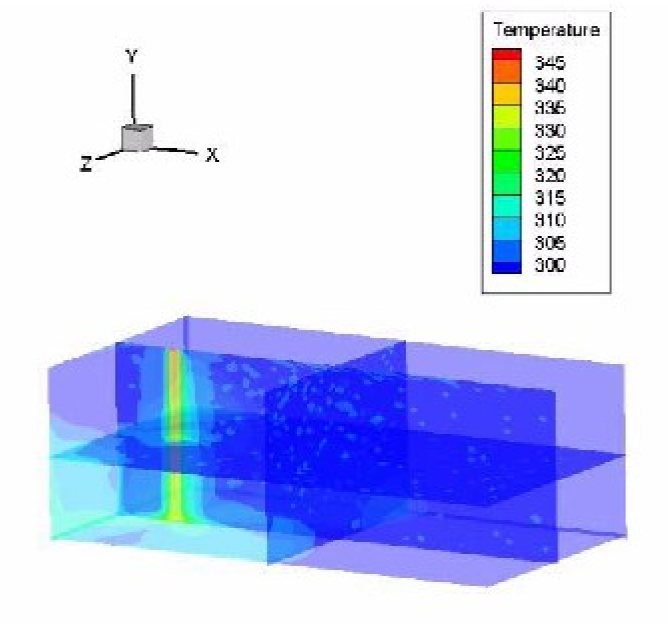


Figure 22: The 3D Model

7.2 Motion Strategy

In the previous section we test the time it takes to be even in temperature in the bathtub without a person's motion. If we turn off the faucet after 35s, the total bath water temperature may become even in 100s. Motion can increase the velocity of the water in bathtub and hence can be used to accelerate the equilibrium time.

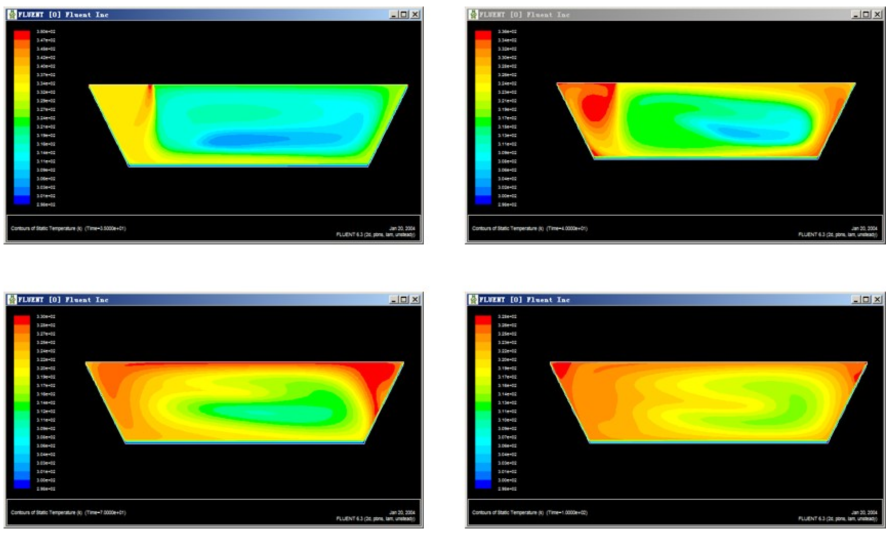


Figure 23: The simulation shows the motions strategy is utterly needed to shorten the even time. We simplified the motions into two type.

We simplified the motions into two type Type one is the up-and-down movement and type two is right-and-left. By choosing the time 1s, 5s, 10s, and 20s we can compare them with each other and obtain the conclusion.

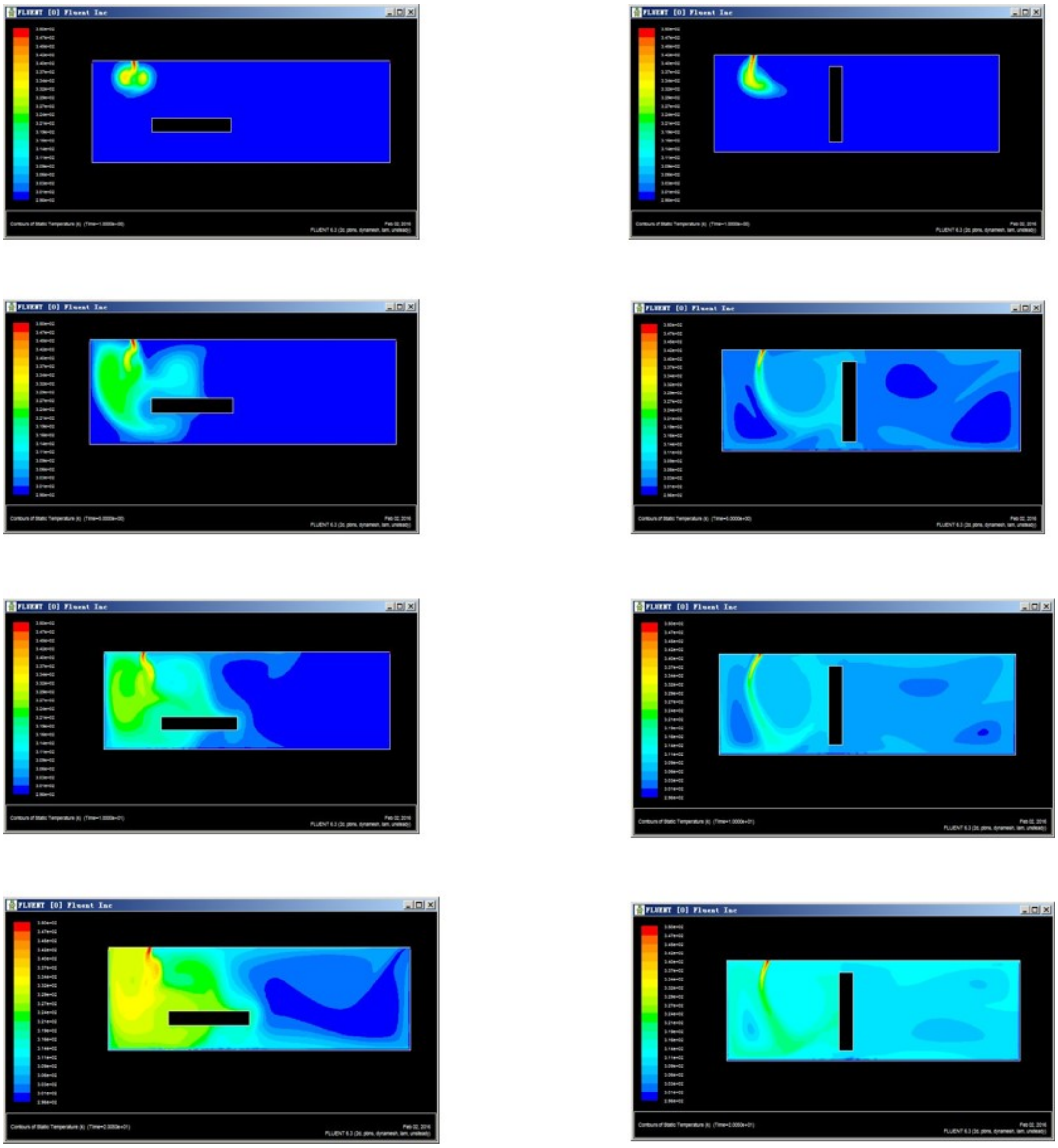


Figure 24: From the figure we can observe that stirring horizontally is more effective than stirring vertically, but both of them distribute faster than the case when there is no stirring

8 Sensitivity Analysis

We change the temperature of the inlet water to obtain different relations between the inlet water velocity and the equilibrium time. We test the temperature 340K, 350K and 360K, and the result show that our model is robust.

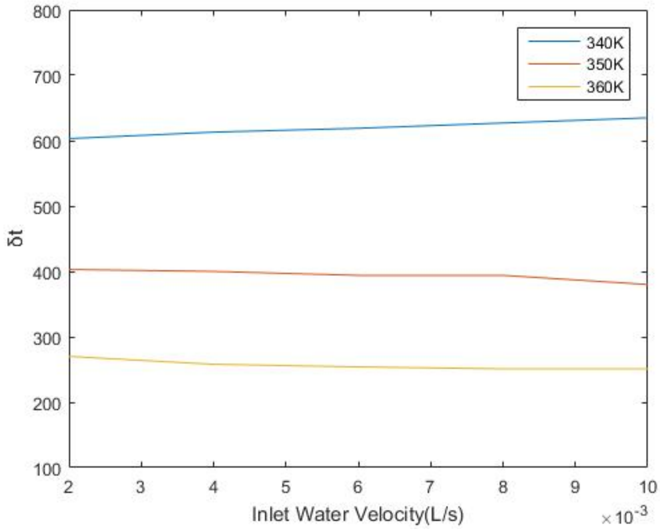


Figure 25: The relation between the equilibrium time and the inlet water velocity does not vary as temperature change, validating that our model is not chaotic.

9 Conclusion

From the above study regarding the shape of the bathtub, the speed and temperature of the inlet-water, the motion strategy, and the bubble bath, we can see that these essential factors may help shorten the even temperature time and to be close to the initial temperature. Eventually, our goal is reached.

10 Strengths and Constraints

10.1 Strengths

1. Visualize the model using Fluent and Matlab.
2. Our model is flexible and can be used to study different variables with just a little variation, including temperature, velocity, shape of the bathtub and so on.

10.2 Constraints

Generally we constructed four prediction models and acquired several observations about the velocities of the fluid based on the *Lattice Boltzmann Model*. However, there are still many questions remains.

1. Qualitative analysis is not enough. Elaborating on the bathtub needs quantitative analysis
2. Need to integrate the temperature factor, because the users want the temperature of the water converge close to the original temperature.
3. Need to extend to a 3D model.
4. Need to add considerations of human agitation and bubbles(which alter the viscosity of the fluid).

′

References

[1] Alexander F J, Chen S, Sterling J D. Lattice boltzmann thermohydrodynamics[J]. Physical Review E, 1993, 47(4): R2249.

[2] Guo Z, Shi B, Zheng C. A coupled lattice BGK model for the Boussinesq equations[J]. International Journal for Numerical Methods in Fluids, 2002, 39(4):325-342.

[3] Kam E W S, So R M C, Leung R C K. Lattice Boltzman Method Simulation of Aeroacoustics and Nonreflecting Boundary Conditions[J]. Aiaa Journal, 2007, 45(7):1703-1712.

[4] Y.B.Bao, J. Meskas, *Lattice Boltzmann Method for Fluid Simulations*, Apr. www.âĂŃcims.âĂŃnyu.âĂŃedu/âĂŃ billbao/âĂŃreport930.âĂŃpdf

[5] D.Bertsimas, J. Tsitsiklis, *Simulated Annealing* Statical Science. 1993.Vol.8.No.1,10-15

[6] Seb, *The lattice Boltzmann method in 25 lines of MATLAB code (LBM)* Matlab Central 14 Oct. 2014 <*Matlab Code*>

[7] J. Montagu *Lattice Boltzmann Matlab Scripts* 4th Earl of Sandwich, exolete <*Matlab Code*>

[8] S. Si *Mathematical modeling algorithm and its applcations* Aug. 2011

[9] Fluent I N C. FLUENT 6.3 userâĂŹs guide[J]. Fluent documentation, 2006.

Here are simulation programmes we used in our model as follow.

Lattice Boltzmann Method Simulation

```
% Modified based on the code of  Iain Haslam, March 2006.
omega=1.0; density=1.0; t1=4/9; t2=1/9; t3=1/36; c_squ=1/3; nx=10; ny=10;
b=94;
c=sqrt(c_squ);
F=repmat(density/9,[nx ny 9]); FEQ=F; msize=nx*ny; CI=[0:msize:msize*7];
T=repmat(300/4,[nx ny 4]);TEQ=repmat(300/4,[nx ny 4]);T0=sum(T,3);
BOUND=zeros(nx,ny);
BOUND(1,2:ny)=1;
BOUND(1:nx,ny)=1;
BOUND(nx,1:ny)=1;
ON=find(BOUND); %matrix offset of each Occupied Node
TO_REFLECT=[ON+CI(1) ON+CI(2) ON+CI(3) ON+CI(4) ...
            ON+CI(5) ON+CI(6) ON+CI(7) ON+CI(8)];
REFLECTED= [ON+CI(5) ON+CI(6) ON+CI(7) ON+CI(8) ...
            ON+CI(1) ON+CI(2) ON+CI(3) ON+CI(4)];
T_TO_REFLECT=[ON+CI(1),ON+CI(2),ON+CI(3),ON+CI(4)];
T_REFLECTED=[ON+CI(3),ON+CI(4),ON+CI(1),ON+CI(2)];
avu=0; prevavu=1; ts=0; deltaU=1e-2; numactivenodes=sum(sum(1-BOUND));
tstop=ts;
thresh=0.2;
stdvalue=0;% std deviation of temp distribution
target=330;
avg=300;
figure;
%stop when (low flow and small std and greater than 100) or longer than
%4000, and the low flow control is the most important
while (ts<40000 && (5e-5<abs((prevavu-avu)/avu)|| stdvalue>thresh ...
|| abs(avg-target)>3) ) || ts<100
      % Propagate (circulation)
    F(:,:,4)=F([2:nx 1],[ny 1:ny-1],4);F(:,:,3)=F(:,[ny 1:ny-1],3);
    F(:,:,2)=F([nx 1:nx-1],[ny 1:ny-1],2);F(:,:,5)=F([2:nx 1],:,5);
    F(:,:,1)=F([nx 1:nx-1],:,1);F(:,:,6)=F([2:nx 1],[2:ny 1],6);
    F(:,:,7)=F(:,[2:ny 1],7); F(:,:,8)=F([nx 1:nx-1],[2:ny 1],8);
    BOUNCEDBACK=F(TO_REFLECT);
    %Densities bouncing back at next timestep
    DENSITY=sum(F,3);
    UX=(sum(F(:,:,[1 2 8]),3)-sum(F(:,:,[4 5 6]),3))./DENSITY;
    UY=(sum(F(:,:,[2 3 4]),3)-sum(F(:,:,[6 7 8]),3))./DENSITY;
    %water flow
  if((avg<target)||(ts<100))
    UX(1,1)=UX(1,1); %Increase inlet pressure
    UY(3,1)=UY(3,1)+deltaU*6;
  else
    UX(1,1)=UX(1,1); %Increase inlet pressure
    UY(3,1)=UY(3,1);
  end

    UX(ON)=0; UY(ON)=0; DENSITY(ON)=0;
    U_SQU=UX.^2+UY.^2; U_C2=UX+UY; U_C4=-UX+UY; U_C6=-U_C2; U_C8=-U_C4;

    %Temperature
    T0PREV=T0;
    T(:,:,1)=T([nx 1:(nx-1)],:,1);
    T(:,:,2)=T(:,[2:ny 1],2);
    T(:,:,3)=T([2:nx 1],:,3);
    T(:,:,4)=T(:,[ny 1:(ny-1)],4);
    T_BOUNCEBACK=T(T_TO_REFLECT);
    T0=sum(T,3);
    %the inlet temperature
    T0(3,1)=350;
    % boundary temperature
    %T0(ON)=300;
    UTX=(sum(T(:,:,1),3)-sum(T(:,:,3),3))./T0;
```

```
    UTY=(sum(T(:,:,2),3)-sum(T(:,:,4),3))./T0;
    TEQ(:,:,1)=T0/4.*(1+2*UX/c);
    TEQ(:,:,2)=T0/4.*(1+2*UY/c);
    TEQ(:,:,3)=T0/4.*(1-2*UX/c);
    TEQ(:,:,4)=T0/4.*(1-2*UY/c);

    % Calculate equilibrium distribution: stationary
    FEQ(:,:,9)=t1*DENSITY.*(1-U_SQU/(2*c_squ));
    % nearest-neighbours
    FEQ(:,:,1)=t2*DENSITY.*(1+UX/c_squ+0.5*(UX/c_squ).^2-U_SQU/(2*c_squ));
    FEQ(:,:,3)=t2*DENSITY.*(1+UY/c_squ+0.5*(UY/c_squ).^2-U_SQU/(2*c_squ))...
    +c*b*(T0-T0PREV)./T0PREV;
    FEQ(:,:,5)=t2*DENSITY.*(1-UX/c_squ+0.5*(UX/c_squ).^2-U_SQU/(2*c_squ));
    FEQ(:,:,7)=t2*DENSITY.*(1-UY/c_squ+0.5*(UY/c_squ).^2-U_SQU/(2*c_squ))...
    -c*b*(T0-T0PREV)./T0PREV;

    % next-nearest neighbours
    FEQ(:,:,2)=t3*DENSITY.*(1+U_C2/c_squ+0.5*(U_C2/c_squ).^2-U_SQU/(2*c_squ));
    FEQ(:,:,4)=t3*DENSITY.*(1+U_C4/c_squ+0.5*(U_C4/c_squ).^2-U_SQU/(2*c_squ));
    FEQ(:,:,6)=t3*DENSITY.*(1+U_C6/c_squ+0.5*(U_C6/c_squ).^2-U_SQU/(2*c_squ));
    FEQ(:,:,8)=t3*DENSITY.*(1+U_C8/c_squ+0.5*(U_C8/c_squ).^2-U_SQU/(2*c_squ));
    F=omega*FEQ+(1-omega)*F;
    T=omega*TEQ+(1-omega)*T;
    F(REFLECTED)=BOUNCEDBACK;
    T(T_REFLECTED)=T_BOUNCEBACK;
    prevavu=avu;avu=sum(sum(UX))/numactivenodes; ts=ts+1;

    %Calculation of degree of enenity
    COLMAT=reshape(T0,nx*ny,1);
    avg=mean(COLMAT);
    stdvalue=std(COLMAT)/avg;
end

colormap(gray(2));image(2-BOUND');hold on;
quiver(2:nx,1:ny,UX(2:nx,:)',UY(2:nx,:)');
title(['Flow field after ',num2str(ts),'\deltat']);xlabel('x');ylabel('y');
```

The Simulated Anneal Model

```
% Modified based on the code of  Iain Haslam, March 2006.
omega=1.0; density=1.0; t1=4/9; t2=1/9; t3=1/36; c_squ=1/3; nx=10; ny=10;
b=94;
c=sqrt(c_squ);
F=repmat(density/9,[nx ny 9]); FEQ=F; msize=nx*ny; CI=[0:msize:msize*7];
T=repmat(300/4,[nx ny 4]);TEQ=repmat(300/4,[nx ny 4]);T0=sum(T,3);
BOUND=zeros(nx,ny);
BOUND(1,2:ny)=1;
BOUND(1:nx,ny)=1;
BOUND(nx,1:ny)=1;
ON=find(BOUND); %matrix offset of each Occupied Node
TO_REFLECT=[ON+CI(1) ON+CI(2) ON+CI(3) ON+CI(4) ...
            ON+CI(5) ON+CI(6) ON+CI(7) ON+CI(8)];
REFLECTED= [ON+CI(5) ON+CI(6) ON+CI(7) ON+CI(8) ...
            ON+CI(1) ON+CI(2) ON+CI(3) ON+CI(4)];
T_TO_REFLECT=[ON+CI(1),ON+CI(2),ON+CI(3),ON+CI(4)];
T_REFLECTED=[ON+CI(3),ON+CI(4),ON+CI(1),ON+CI(2)];
avu=0; prevavu=1; ts=0; deltaU=1e-2; numactivenodes=sum(sum(1-BOUND));
tstop=ts;
thresh=0.2;
stdvalue=0;% std deviation of temp distribution
target=330;
avg=300;
figure;
```

```
%stop when (low flow and small std and greater than 100) or longer than
%4000, and the low flow control is the most important
while (ts<40000 && (5e-5<abs((prevavu-avu)/avu)|| stdvalue>thresh ...
|| abs(avg-target)>3) ) || ts<100
      % Propagate (circulation)
    F(:,:,4)=F([2:nx 1],[ny 1:ny-1],4);F(:,:,3)=F(:,[ny 1:ny-1],3);
    F(:,:,2)=F([nx 1:nx-1],[ny 1:ny-1],2);F(:,:,5)=F([2:nx 1],:,5);
    F(:,:,1)=F([nx 1:nx-1],:,1);F(:,:,6)=F([2:nx 1],[2:ny 1],6);
    F(:,:,7)=F(:,[2:ny 1],7); F(:,:,8)=F([nx 1:nx-1],[2:ny 1],8);
    BOUNCEDBACK=F(TO_REFLECT);
    %Densities bouncing back at next timestep
    DENSITY=sum(F,3);
    UX=(sum(F(:,:,[1 2 8]),3)-sum(F(:,:,[4 5 6]),3))./DENSITY;
    UY=(sum(F(:,:,[2 3 4]),3)-sum(F(:,:,[6 7 8]),3))./DENSITY;
    %water flow
  if((avg<target)||(ts<100))
    UX(1,1)=UX(1,1); %Increase inlet pressure
    UY(3,1)=UY(3,1)+deltaU*6;
  else
    UX(1,1)=UX(1,1); %Increase inlet pressure
    UY(3,1)=UY(3,1);
  end

    UX(ON)=0; UY(ON)=0; DENSITY(ON)=0;
    U_SQU=UX.^2+UY.^2; U_C2=UX+UY; U_C4=-UX+UY; U_C6=-U_C2; U_C8=-U_C4;

    %Temperature
    T0PREV=T0;
    T(:,:,1)=T([nx 1:(nx-1)],:,1);
    T(:,:,2)=T(:,[2:ny 1],2);
    T(:,:,3)=T([2:nx 1],:,3);
    T(:,:,4)=T(:,[ny 1:(ny-1)],4);
    T_BOUNCEBACK=T(T_TO_REFLECT);
    T0=sum(T,3);
    %the inlet temperature
    T0(3,1)=350;
    % boundary temperature
    %T0(ON)=300;
    UTX=(sum(T(:,:,1),3)-sum(T(:,:,3),3))./T0;
    UTY=(sum(T(:,:,2),3)-sum(T(:,:,4),3))./T0;
    TEQ(:,:,1)=T0/4.*(1+2*UX/c);
    TEQ(:,:,2)=T0/4.*(1+2*UY/c);
    TEQ(:,:,3)=T0/4.*(1-2*UX/c);
    TEQ(:,:,4)=T0/4.*(1-2*UY/c);

    % Calculate equilibrium distribution: stationary
    FEQ(:,:,9)=t1*DENSITY.*(1-U_SQU/(2*c_squ));
    % nearest-neighbours
    FEQ(:,:,1)=t2*DENSITY.*(1+UX/c_squ+0.5*(UX/c_squ).^2-U_SQU/(2*c_squ));
    FEQ(:,:,3)=t2*DENSITY.*(1+UY/c_squ+0.5*(UY/c_squ).^2-U_SQU/(2*c_squ))...
    +c*b*(T0-T0PREV)./T0PREV;
    FEQ(:,:,5)=t2*DENSITY.*(1-UX/c_squ+0.5*(UX/c_squ).^2-U_SQU/(2*c_squ));
    FEQ(:,:,7)=t2*DENSITY.*(1-UY/c_squ+0.5*(UY/c_squ).^2-U_SQU/(2*c_squ))...
    -c*b*(T0-T0PREV)./T0PREV;

    % next-nearest neighbours
    FEQ(:,:,2)=t3*DENSITY.*(1+U_C2/c_squ+0.5*(U_C2/c_squ).^2-U_SQU/(2*c_squ));
    FEQ(:,:,4)=t3*DENSITY.*(1+U_C4/c_squ+0.5*(U_C4/c_squ).^2-U_SQU/(2*c_squ));
    FEQ(:,:,6)=t3*DENSITY.*(1+U_C6/c_squ+0.5*(U_C6/c_squ).^2-U_SQU/(2*c_squ));
    FEQ(:,:,8)=t3*DENSITY.*(1+U_C8/c_squ+0.5*(U_C8/c_squ).^2-U_SQU/(2*c_squ));
    F=omega*FEQ+(1-omega)*F;
    T=omega*TEQ+(1-omega)*T;
    F(REFLECTED)=BOUNCEDBACK;
```

```
    T(T_REFLECTED)=T_BOUNCEBACK;
    prevavu=avu;avu=sum(sum(UX))/numactivenodes; ts=ts+1;

    %Calculation of degree of enenity
    COLMAT=reshape(T0,nx*ny,1);
    avg=mean(COLMAT);
    stdvalue=std(COLMAT)/avg;
end

colormap(gray(2));image(2-BOUND');hold on;
quiver(2:nx,1:ny,UX(2:nx,:)',UY(2:nx,:)');
title(['Flow field after ',num2str(ts),'\deltat']);xlabel('x');ylabel('y');
```

```
%* Modified based on
%* Date:2012č■11-29
%* Author:steven
%* Email:hxs2004@126.com


clear;
clc;
XMAX= 30;
YMAX = 0.1;

MarkovLength = 10000;
DecayScale = 0.95;
StepFactor = 0.02;
Temperature=30;
Tolerance = 1*1e-8;
AcceptPoints = 0.0;
rnd =rand;
PreX = XMAX * rand ;
PreY = YMAX * rand;
PreBestX  = PreX;
PreBestY = PreY;
PreX = XMAX * rand ;
PreY = YMAX * rand;
BestX = PreX;
BestY = PreY;
mm=abs( ObjectFunction( BestX,BestY)-ObjectFunction (PreBestX, PreBestY));
disp('mm');
disp(mm);
ii = 0;
disp(BestX);
disp(BestY);
while mm > Tolerance && ii < 100

Temperature=DecayScale*Temperature;
AcceptPoints = 0.0;
for i=0:MarkovLength:1
p=0;
while p==0
     NextX = PreX + StepFactor*XMAX*(rand-0.5);
    NextY = PreY + StepFactor*YMAX*(rand-0.5);
    if p== (~(NextX >= -XMAX && NextX <= XMAX && NextY >= -YMAX && NextY <= YMAX))
        p=1;
    end
end


if (ObjectFunction(BestX,BestY) > ObjectFunction(NextX,NextY))
PreBestX =BestX;
```

```
PreBestY = BestY;
BestX=NextX;
BestY=NextY;
end


if( ObjectFunction(PreX,PreY) - ObjectFunction(NextX,NextY) > 0 )
PreX=NextX;
PreY=NextY;
AcceptPoints=AcceptPoints+1;
else
 changer = -1 * ( ObjectFunction(NextX,NextY) - ObjectFunction(PreX,PreY) ) / Temperature ;
rnd=rand;
p1=exp(changer);
double (p1);

if p1 > rand
    PreX=NextX;
    PreY=NextY;
    AcceptPoints=AcceptPoints+1;
end
end
end
    mm=abs( ObjectFunction( BestX,BestY)-ObjectFunction (PreBestX, PreBestY));
    ii = ii + 1;
end
str =[ ’ŒîÐąÖţÔÚţã:(’, num2str(BestX+330), ’ ’, num2str(BestY), ’)’];
disp(str);
[time,finaltemp]=ObjectFunction(BestX, BestY)
```