For office use only	Team Control Number	For office use only
T1 ________________	**19064**	F1 ________________
T2 ________________		F2 ________________
T3 ________________	Problem Chosen **A**	F3 ________________
T4 ________________		F4 ________________

2013
Mathematical Contest in Modeling (MCM/ICM) Summary Sheet

.

In this paper we propose a mathematical model to optimize the pan shape for a rectangular Oven in order to have most pans inside and evenly distributed heat pattern for each pan.

Our approach is based upon two methodologies:

- Transient Finite Difference Equations: It generates heat pattern variation with time for any shape by dividing them into infinitesimally dense grid and performing numerical analysis at nodal points. The key application of this assumption is that it can be easily computed through simulations.
- Shape Variation and Packaging: The variation of shape from rectangle to circle for the pans is achieved by continuously increasing the boundary corners (starting from 4 in a rectangle) with change in the width to length ratio of the figure. This leads to conversion of shapes into vector forms that could be easily plotted and examined. These shapes are then approximated by bounding rectangle to apply Recursive Partitioning Algorithm for packing and the variation of total no. of pans with shape is obtained.

The evenness of the heat distribution is defined as a function of Temperature gradients across its outer boundary.

The above model is simulated and the variation of heat distribution along the periphery is realized. The distribution obtained with the shape variation defines approximate functions for evenness of heat distribution and maximum no. of pans packed. These functions are combined with a weight value of p and 1-p. This function is maximized to obtain best shape forms for different oven width to length ratios, evenness of heat distribution and no. of pans contained.

An advertisement and lots of code at the end.

The Ultimate Brownie Pan

Team# 19064

Introduction

The Heat Distribution in an Oven Pan is not even due to difference in heat transfer mechanisms at corners, edges and points inside of the Pan. The points of the pan exposed directly to the heat obtain heat through convection as supposed to conduction through the adjacent points inside the Pan. Hence, for a rectangular pan, the product gets overcooked in the corners (and less across edges), while for a round pan the product is not overcooked at the edges. Other than the even heat distribution requirement, the rectangular shape of most of the Ovens encourages the usage of the Oven area to include maximum no. of pans. Packing of round pans inside a rectangular area is quite inefficient due to larger spacing in between.

It is required that the even heat distribution of each pan (with a weight p assigned to it) along with the maximization of the no. of pans (with a weight 1-p assigned to it) inside a given rectangular oven should be attained by deciding the most efficient shape ranging from a rectangle to round pan. Also width to length ratio for the oven (W/L) could be responsible for the most efficient shape determination for the pans. In this paper, a model is proposed to determine the best type for the shape of pan with given W/L Oven ratio, Pan Area A_P and Oven Area (N_OA) by implementing "Transient 2D finite difference equations" to obtain heat distribution and Recursive Partitioning Approach to maximize no. of pans.

It is required that we first assume a heat environment across the pan. Then, by considering the heat transfer mechanisms, heat pattern around the Pan can be obtained with variation across Time.

It is also required that the shape variation from rectangle to circle are obtained in a manner such that the heat distribution gets more even with the deviation of the shape from rectangle towards circle. These shapes generated should be easily recognizable in available solutions for packaging problems.

These data would be then simulated into MATLAB to obtain following distributions:

- Heat Distribution (H) as a function of Shapes (rectangle to circle)
- Maximum no. of Pans (N) inside given (W/L) Rectangular Oven as a function of Shapes
- Assign a weight p and 1-p to H and N, and vary p to obtain Optimum Shape as function of p.
- Variation of Optimum shape as function of (W/L)

- Vary the (W/L) ratio and p to obtain Optimum Shapes.

An analysis of these variations can help determine the best shape or the range to be considered for the best shape.

Assumptions

In our model we will deal with the following parameters:

Table 1: Parameter Descriptions

Symbols	Description
A_P	Area of the Pan
N_O	Oven to Pan Area ratio
C_O	Width/Length ratio (W/L) for the Oven
N	Maximum no. of Pans that can fit in an Oven
T_P	Initial Temperature of the Pan
T_∞	Temperature Outside the Pan
a=l/2	Major axis length of equivalent ellipse
b=w/2	Minor axis length of equivalent ellipse (a>=b ; always)
c	=b/a (ratio of axis lengths) ; A Shape Defining Parameter
n	Total no. of Corners in the shape; A Shape Defining Parameter
L,W	Length , Width of Oven
R_O	Ratio of width to length for Oven
H	Even Heat Distribution
p	Weight given to H for Shape optimization
$\boldsymbol{t_f}$	Time taken for the product to get cooked

Following assumptions are taken to implement the model that we have constructed later:

- It is assumed that there is a constant temperature across the pan ($T_\infty=200°C$). This assumption means that first the heat inside the oven was developed and set to constant and only after that the pan started its heating process from the initial room temperature ($T_P=30°C$). This error is negligible if the final heating temperature inside the oven (around the pan) required to heat the material is attained quite fast.

- The pan is assumed as a 2D model with the convection taking place across the edges and the corners. Convection along the third dimension (top and bottom) is not taken into consideration since its effect on each point in the 2D Plane is assumed to be almost uniform.

- During the transformation of shapes from rectangle to circle, the 'Finite difference equations' are applied with an assumption that the curvature shapes at the edges are rectangular with very small rectangular dimensions.

- While considering the optimized Packing of shapes in the rectangular Oven, the Recursive Partitioning Algorithm is applied to each shape by considering each shape as rectangular with dimensions (*2a*2b*). This also means that for each pan, ***n*** is a multiple of 4.This can be considered adequate since packaging problems have no definite solution. Shapes considered this way can still find an efficient solution for some C_O.

- Initially some parameters are assumed to be constant. Their values are obtained from applied cases and set as below:

A_P= 600cm^2 ; N_O = 25; C_O = 2/3 ; T_∞=200°C ; T_P=30°C ;

Shape Variation

The shapes are supposed to be varied from rectangle to a circle. In our model, we assume two parameters ***n*** and ***c*** to define each shape [Table 1] such that,

$$n \in N - \{1,2,3\} ;\ c \in (0,1]$$

Thus, Rectangle is (n=4), Circle is (n= ∞ , c=1) & Square is (n=4, c=1)

Since the Area of each shape is assumed to be constant i.e. A_P and the shape is dependent on ***c*** and ***n***, the ***a*** and ***b*** are determined as a function of ***c,n*** and ***A_P***. The change of the shape is modeled in the following format:

- Draw an ellipse with ***b/a =c***

$$(\frac{x}{a})^2 + (\frac{y}{b})^2 = 1$$

- Divide it in *n* points as ***(a*cos(m*pi/n), c*a*sin(m*pi/n))***. Draw tangents to each of these points. The figure enclosing these tangents is the required shape for given ***c,n***
- Equate the Area of this figure with A_P and the following relationship is obtained:

$$a = \sqrt{\frac{A_P}{c * n * \tan(\frac{\pi}{n})}}\ , b = c * a = \sqrt{\frac{c * A_P}{n * \tan(\frac{\pi}{n})}}$$

Now starting from ***n=4*** and some ***c***, if the ***n*** becomes very large, the shape corresponds to an ellipse of area A_P. And if ***c*** becomes unity, the shape becomes a circle of area A_P.

Once a system for variation of shapes is developed, the heat distribution across these shapes and packaging for each shape can be obtained. We can plot ***H*** and ***N*** as a function of ***(n,c)*** to implement the shape variation of pan.

For Shape***(c,n)*** , the coordinates for ***n***-points are:

$$x_m = \frac{a\cos(\frac{(2m-1)\pi}{n})}{\cos(\frac{\pi}{n})}, y_m = \frac{b\sin(\frac{(2m-1)\pi}{n})}{\cos(\frac{\pi}{n})},$$

where, ***m*** ϵ {1,2,…..,n}

These coordinate equations would be helpful in computational purposes [Appendix].

Heat Distribution

Pan of the oven is considered as two dimensional object as we are interested in the temperature distribution on its surface only. Heat transfer inside the pan occurs by conduction. Heat transfer between air and pan occurs by convection.

Instead of finding temperature of every point, a numerical technique called finite-difference method is used for determination of temperature only at discrete points. In this method whole pan is divided into grid and numerical analysis is done on nodal points.

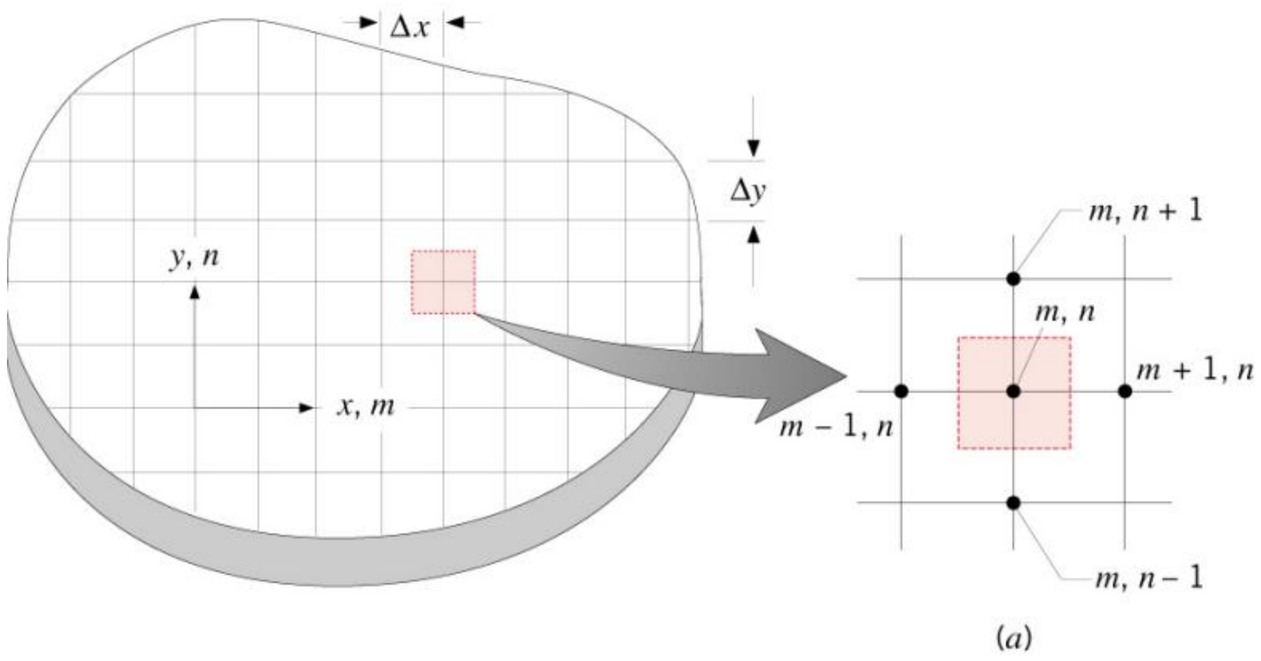


Figure 1

Initial temperature of whole pan is taken as 300C. After putting it in the oven containing hot air at temperature 2000C which is uniform across oven, transient of temperature is obtained using finite-difference equations shown in fig 2.

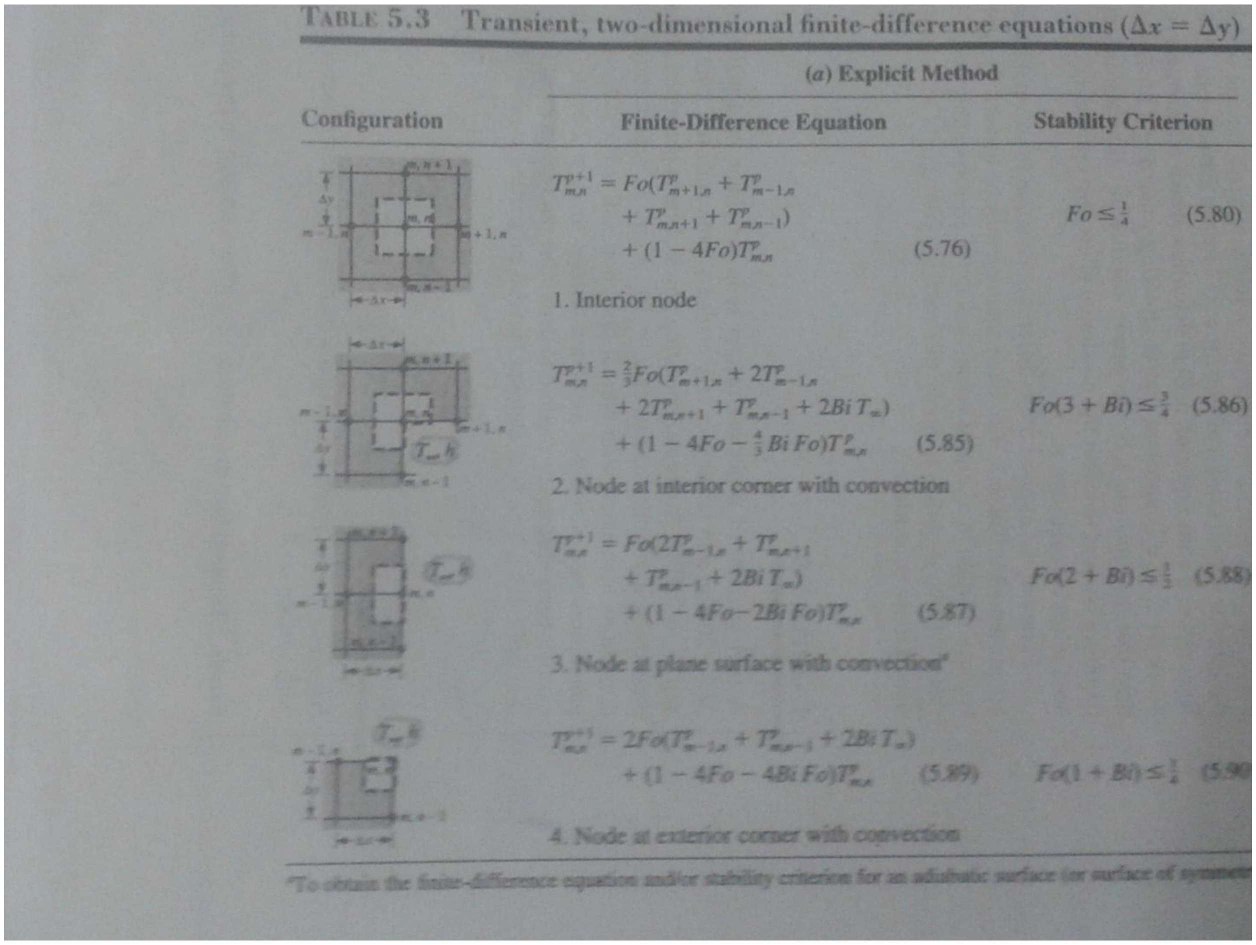


TABLE 5.3 Transient, two-dimensional finite-difference equations ($\Delta x = \Delta y$)

Configuration	(a) Explicit Method			
	Finite-Difference Equation		Stability Criterion	
	$T_{m,n}^{p+1} = Fo(T_{m+1,n}^{p} + T_{m-1,n}^{p} + T_{m,n+1}^{p} + T_{m,n-1}^{p}) + (1 - 4Fo)T_{m,n}^{p}$ 1. Interior node	(5.76)	$Fo \leq \frac{1}{4}$	(5.80)
	$T_{m,n}^{p+1} = \frac{2}{3}Fo(T_{m+1,n}^{p} + 2T_{m-1,n}^{p} + 2T_{m,n+1}^{p} + T_{m,n-1}^{p} + 2Bi\,T_{\infty}) + (1 - 4Fo - \frac{4}{3}Bi\,Fo)T_{m,n}^{p}$ 2. Node at interior corner with convection	(5.85)	$Fo(3 + Bi) \leq \frac{3}{4}$	(5.86)
	$T_{m,n}^{p+1} = Fo(2T_{m-1,n}^{p} + T_{m,n+1}^{p} + T_{m,n-1}^{p} + 2Bi\,T_{\infty}) + (1 - 4Fo - 2Bi\,Fo)T_{m,n}^{p}$ 3. Node at plane surface with convection[a]	(5.87)	$Fo(2 + Bi) \leq \frac{1}{2}$	(5.88)
	$T_{m,n}^{p+1} = 2Fo(T_{m-1,n}^{p} + T_{m,n-1}^{p} + 2Bi\,T_{\infty}) + (1 - 4Fo - 4Bi\,Fo)T_{m,n}^{p}$ 4. Node at exterior corner with convection	(5.89)	$Fo(1 + Bi) \leq \frac{1}{4}$	(5.90

[a]To obtain the finite-difference equation and/or stability criterion for an adiabatic surface (or surface of symmetr

Figure 2

Using these equations temperature on discrete time instances can be computed. Out of two methods Explicit method is used which computes temperature at time instance (p+1)Δt using temperature values at time instance pΔt. Where p is an integer. Temperature profile of pan at $\boldsymbol{t_f}$ = 5s is obtained using its initial temperature.

$$\text{Where } F_{0}=\frac{\alpha\,\Delta t}{(\Delta x)^2}\quad Bi=\frac{h\Delta x}{k}$$

α=120e-6 m2/s

h= 200 W/m2 .K when air is under force convection

k=50 W/m.K

<u>**Determining Evenness of Heat Distribution**</u>

In order to determine the evenness of heat distribution, we use the ***H vs (c,n)*** Graph and calculate the minimum, maximum and standard deviation of the Temperature along the boundary of the shape.

Evenness of Heat Distribution (***H***) is defined as

$$H = \frac{1}{\sigma(c,n)}$$

where, ***H*** is unit less and $\sigma(c,n)$ is the standard deviation along the boundary of the shape.

This measure of evenness would be appropriate since our penultimate aim is to have the similar distribution of temperature at boundaries after $\boldsymbol{t_f}$ time. This boundary turns out to be equal to Δx. In this paper, we set $\Delta x = 0.5\text{cm}$.

Packaging of Pans

For each shape that we are considering in the ***(c,n)*** domain, we need to obtain the maximum no. of shapes that can be packed without overlap into the ***LxW*** rectangular Oven. The Area of the Oven is $\boldsymbol{N_O A_P}$. An effective method to pack rectangles of ***l x w*** in ***L x W*** is through Recursive Partitioning Algorithm. So, if we consider each shape as a rectangle of some length ***l*** and width ***w*** , then this algorithm would be easily applicable. For simplicity we consider shapes ***(c,n)*** to be confined only for ***n*** that are multiples of 4. This is because for multiples of 4, a rectangle of length ***l=2a*** and ***w=2b*** can be considered for packaging.

The Recursive Partitioning algorithm is combination of Recursive five-block algorithm and L-block algorithm. The maximum of the two algorithm output is supposed to be the optimal method of arranging rectangles inside rectangles either horizontally or vertically.

The code iterated over a defined vector space of ***(c,n)*** leads to construction of $\boldsymbol{N(c,n,R_O)}$ vs ***(c,n).***

Optimization of Shape

After getting evenness as a function H(n,c) and maximum pan as a function $N(n,c,R_o)$ optimized shape S(n,c) for given p can be obtained by maximizing following function

$$P*H(n,c) + (1-p) *N(n,c,R_o)$$

by taking n and c as independent variables. Where p is weight given to evenness and 1-p is weight given to maximum number of pans.

From this we get n and c as function of p and R_o. Finally we get shape $S(n(p, R_o), c(p, R_o))$ which represents variation in shape for different p and $R_o = W/L$.

Model Testing

Simulation results:

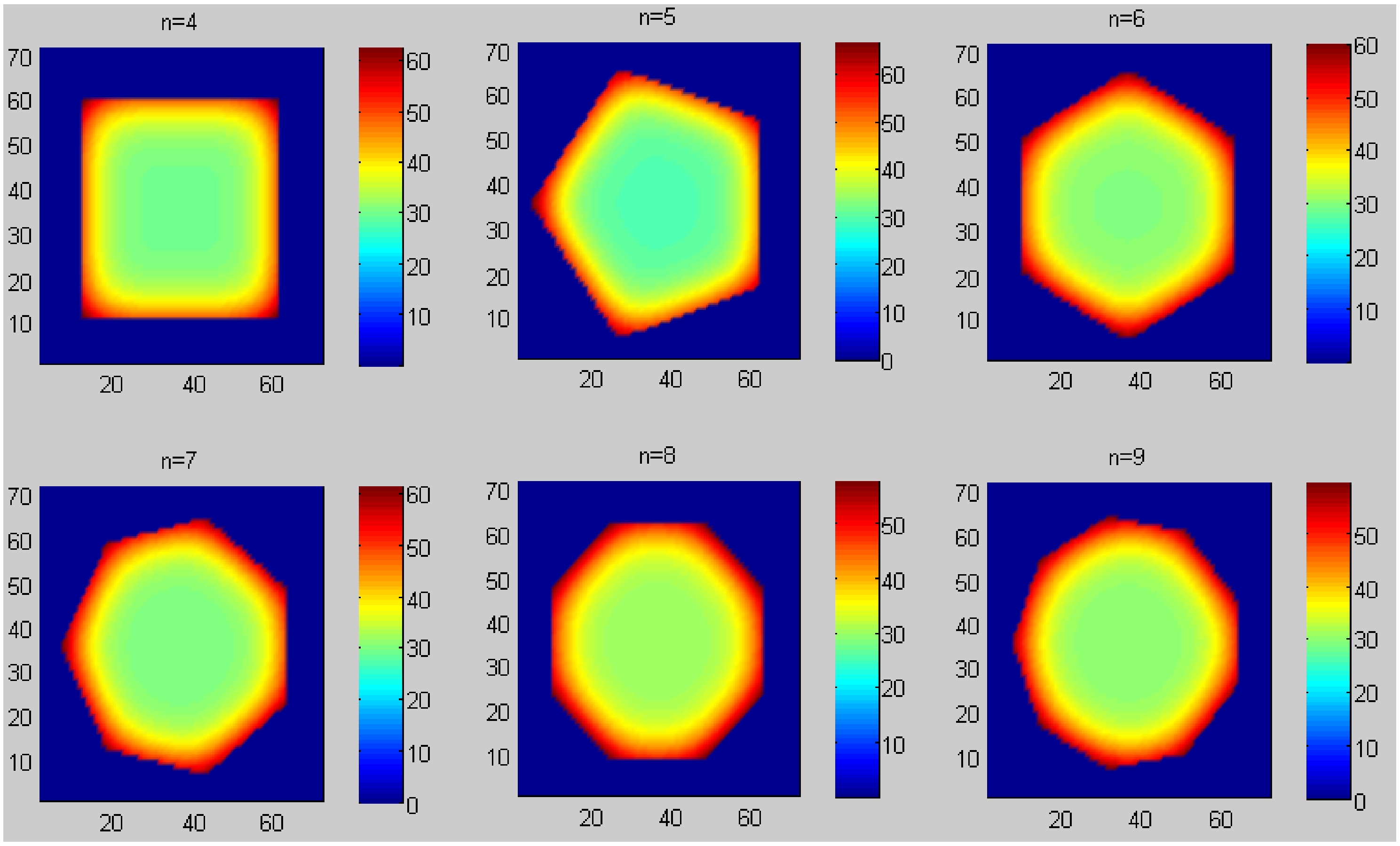


Figure 3

The figure above depicts the heat distribution, the temperature at each point within a number of regular polygon shapes starting from a square (4 sides) to a nonagon (9 sides). It can be seen that according to our model, the heat distribution along the edges is more uniform as the number of sides increases.

Any region outside the polygon is represented in blue. Higher the temperature of the region, the color is more toward red. The temperature distribution is obtained by simulating 10 seconds of heating with our model. The number of sides is denoted above each sub-figure (n=i, i=4, 5, …, 9).

It can also be observed from the color scale to the right of each sub-figure that the maximum temperature in the polygon which at occurs at the corners also decreases as the number of sides increases.

Simulation results:

The results of a simulation to determine a degree of evenness in a region along the outer edge of a pan is presented below.

The initial temperature distribution is obtained by simulating 5 seconds of heating.

Polygons considered with c = 0.8, n = 2 to 24

Various parameters that could be considered are Minimum temperature within the region, Maximum temperature within the region of the pan, Std. deviation of Temperatures within the region, Average of Temperatures within the region.

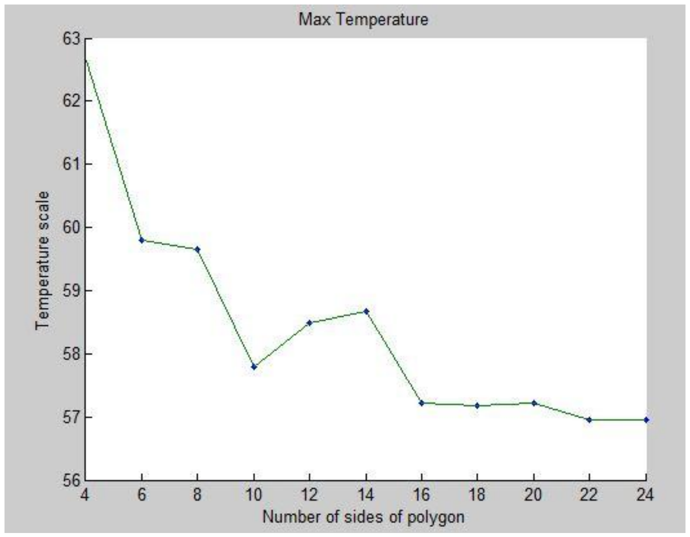


Figure 4

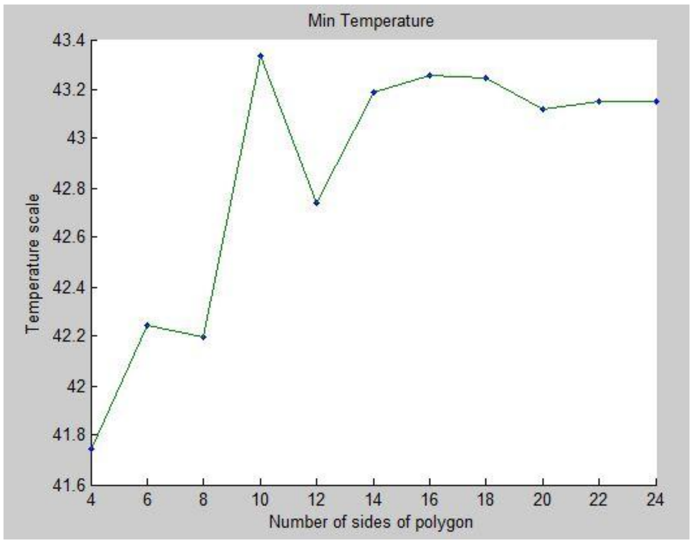


Figure 5

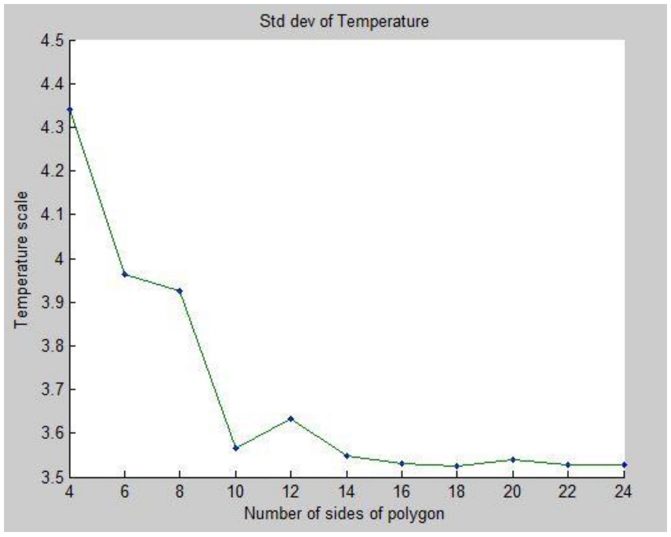


Figure 6

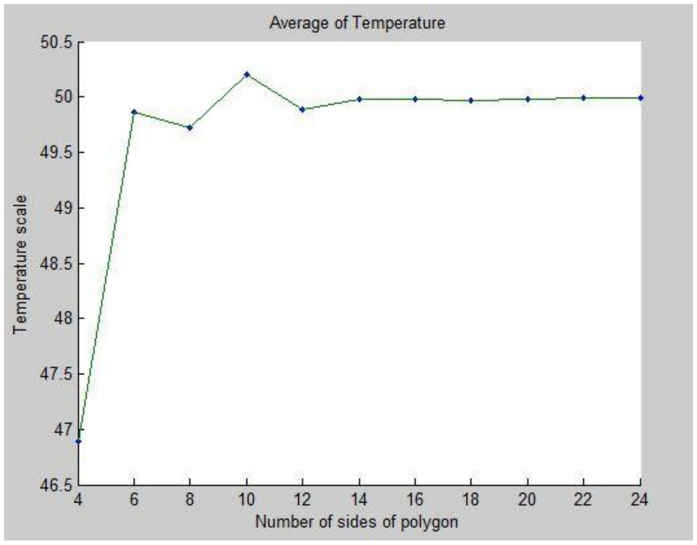


Figure 7

Max temperature decreases with increasing number of sides. Min temperature increases with increasing number of sides. Std. dev. of temperatures is decreases with increasing number of sides.

The trends observed are consistent with the expectations that uniformity of heat distribution increases as number of sides increases. These observations also suggest that the parameters considered above can be used to measure evenness. The average of temperatures is almost constant within the region considered and is not considered for evenness.

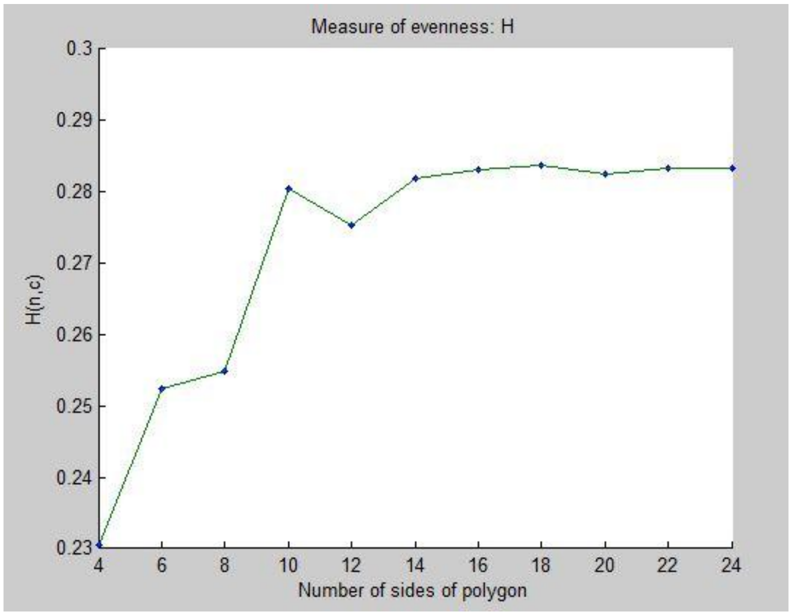


Figure 8

After considering the various parameters, the measure of evenness H(n,c) is modeled sufficiently by the inverse of std. dev. of temperatures. It is also possible to include the other parameter as min and max temp also.

NOTE:

The region of the pan over which the preceding analysis is performed is a annular region along the outer edge of fixed thickness 15mm for all shapes (n,c). The values of temperature are determined from simulations of 5 seconds of heating of the pan.

Limitations

- Considering the shapes as rectangles while packing and the algorithm used leads to a lot of packing inefficiency. Considering only vertical and horizontal arrangements also leads to a lot of the error.

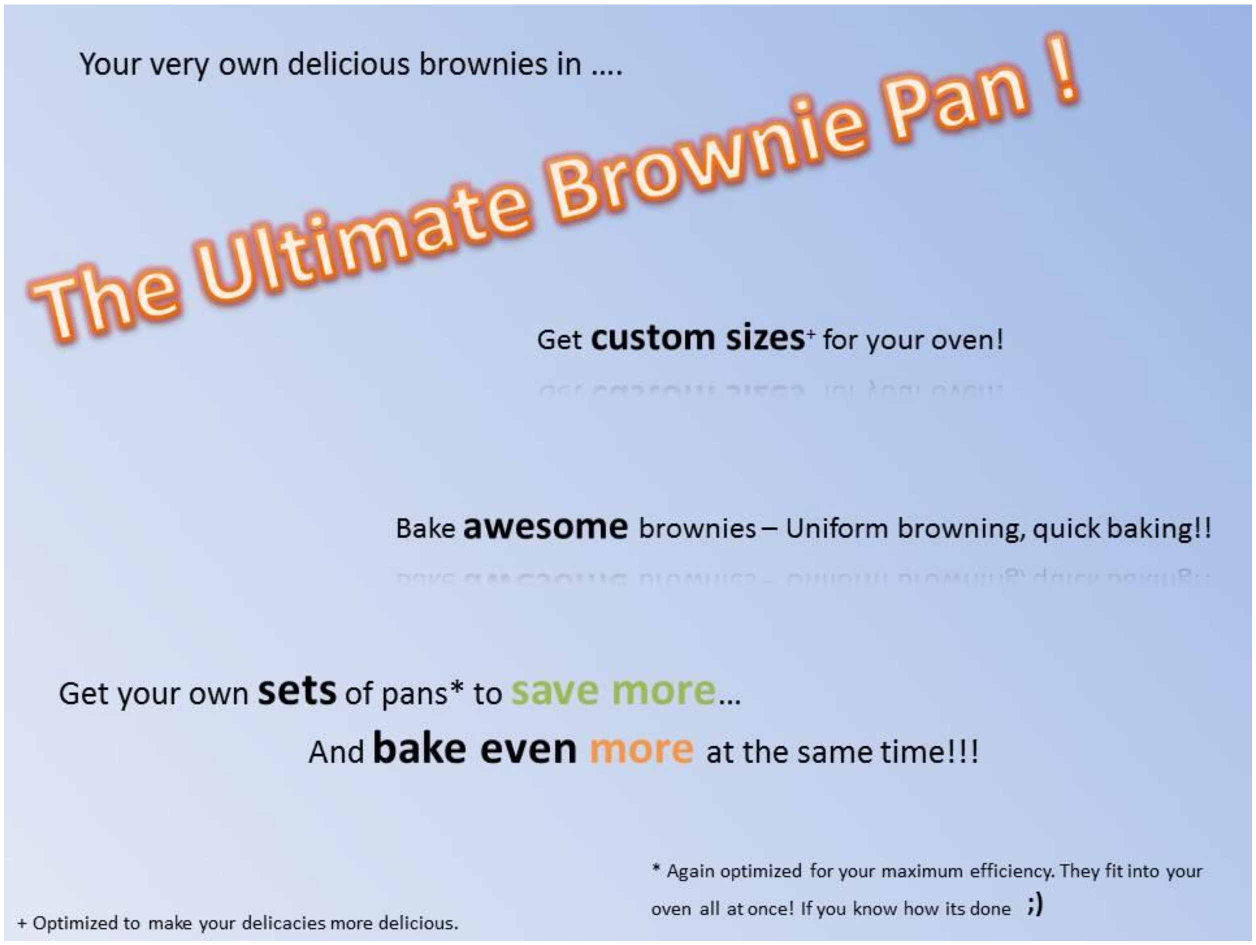
References

E. G. Birgin, R. D. Lobato and R. Morabito. An effective recursive partitioning approach for the packing of identical rectangles in a rectangle. Journal of the Operational Research Society, (2010) 61, 306–320. (DOI: 10.1057/jors.2008.141)

F. Incropera, D. Dewitt, T. Bergman, A. Lavine. Fundamentals of Heat and Mass Transfer. 6th edition, (2011) 201-340

http://lagrange.ime.usp.br/~lobato/packing/

Advertisement:



Your very own delicious brownies in ....

The Ultimate Brownie Pan !

Get **custom sizes**+ for your oven!

Bake **awesome** brownies – Uniform browning, quick baking!!

Get your own **sets** of pans* to **save more**...

And **bake even more** at the same time!!!

+ Optimized to make your delicacies more delicious.

* Again optimized for your maximum efficiency. They fit into your oven all at once! If you know how its done ;)

Advertisement for the Brownie Gourmet Magazine.

Matlab Code:

```
%% Start
clc;
clear;
close all;

%% Dim
W=0.55;  % max b
L=0.55;  % max a

dx=0.005;
dy=0.005;

m=L/dx;
n=W/dy;

A=600/(dx*dy*100*100);
c=0.5;

step=4;
NN=20;   % max corners

TT=zeros(m+2,n+2,NN/step-3);
PP=zeros(m+2,n+2,NN/step-3);

%% Thermal

T0=30;  % room temp
Tinf=200;

h=200;
k=50;
Bi=h*dx/k;
alpha=0.000120;
F=1/(4*(1+Bi));
dt=F*(dx*dy)/alpha;
dt=dt*0.99;

F=alpha*dt/(dx*dy);

t=dt;
tkon=5;

%% Corners loop
for nn=4:step:NN

    T=ones(m+2,n+2)*T0;
    T(1,:)=0;
    T(m+2,:)=0;
    T(:,1)=0;
    T(:,n+2)=0;
    T1=T*0;
```

```
P=ones(size(T));

N=nn;
a=sqrt(A/(c*N*tan(pi/N)));
b=sqrt(A*c/(N*tan(pi/N)));

px=m+2;
py=n+2;

l=1:N;
Xm=a*cos((2*l-1)*pi/N)/cos(pi/N);
Ym=b*sin((2*l-1)*pi/N)/cos(pi/N);

%% Shape

for l=1:N
    if(l==1)
        f0=0-Ym(l)+Xm(l)*(Ym(l)-Ym(N))/(Xm(l)-Xm(N));

        for i=-px/2+1:px/2
            for j=-py/2+1:py/2
                xt=j;
                yt=i;
                fx=(yt-Ym(l))-(xt-Xm(l))*(Ym(l)-Ym(N))/(Xm(l)-Xm(N));
                if(fx*f0<0)
                    P(i+px/2,j+py/2)=-1;
                end
            end
        end
    else
        f0=0-Ym(l)+Xm(l)*(Ym(l)-Ym(l-1))/(Xm(l)-Xm(l-1));

        for i=-px/2+1:px/2
            for j=-py/2+1:py/2
                xt=j;
                yt=i;
                fx=(yt-Ym(l))-(xt-Xm(l))*(Ym(l)-Ym(l-1))/(Xm(l)-Xm(l-1));
                if(fx*f0<0)
                    P(i+px/2,j+py/2)=-1;
                end
            end
        end
    end
end

sum=P*0-8;
for i=2:px-1
    for j=2:py-1
        if(P(i,j)==1)
            sum(i,j)=P(i-1,j)+P(i+1,j)+...
                    +P(i,j+1)+P(i,j-1)+...
                    +P(i-1,j+1)+P(i-1,j-1)+...
                    +P(i+1,j+1)+P(i+1,j-1);
        end
    end
    end

    boundary=zeros(m*n,2);
    bc=1;

    P1=P;

    %% Boundary
    for i=2:px-1
        for j=2:py-1
            if(P(i,j)==1)
                if(sum(i,j)<=7&&sum(i,j)~=-2)
                    P1(i,j)=0;
                    boundary(bc,:)=[i j];
                    bc=bc+1;

                elseif(sum(i,j)==-2&&P(i-1,j)+P(i+1,j)+...
                    +P(i,j+1)+P(i,j-1)~=0)
                    P1(i,j)=-1;
                elseif(sum(i,j)==-2&&P(i-1,j)+P(i+1,j)+...
                    +P(i,j+1)+P(i,j-1)==0)
                    P1(i,j)=0;
                end
            end
        end
    end
    P=P1;
    bc=bc-1;
    boundary(1:bc,:);
    PP(:,:,nn/step)=P;
%     figure;
%     pcolor(P)
%     colorbar

%     pause;
    %% Model
    V=T.*(P+1);

    sum1=P*0;

    px=m+2;
    py=n+2;
    for i=2:px-1
        for j=2:py-1
            sum1(i,j)=P(i-1,j)+P(i+1,j)+...
                    +P(i,j+1)+P(i,j-1);
        end
    end

    sum=0;

    %% Time
    t=dt;
    while t<tkon
        for i=1:m
```

```
                for j=1:n
                    py=j+1;
                    px=i+1;

                    sum=sum1(px,py);
                    v=V(px-1,py)+V(px+1,py)+V(px,py+1)+V(px,py-1);

                    if(P(px,py)==1)
                            T1(px,py)=F*(T(px-
1,py)+T(px+1,py)+T(px,py+1)+T(px,py-1))+(1-4*F)*T(px,py);
                    elseif(P(px,py)==0)
                        switch (sum)
                            case 2
                                T1(px,py)=2/3*F*(v+2*Bi*Tinf)+(1-4*F-
4/3*Bi*F)*T(px,py);
                            case 0
                                T1(px,py)=F*(v+2*Bi*Tinf)+(1-4*F-
2*Bi*F)*T(px,py);
                            case -2
                                T1(px,py)=2*F*(v+2*Bi*Tinf)+(1-4*F-
4*Bi*F)*T(px,py);
                        end
                    end
                end
            end

            T=T1;

            V=T.*(P+1);

            clc;
            nn
            t=t+dt

%           figure;
%       pcolor(T(1:m+1,1:n+1));
%       shading interp;
%       retazec=sprintf('Time=%g s',t);
%       title(retazec);
%       colorbar;
%       pause(0.002);
        end

        %     T
        figure;
        pcolor(T(1:m+1,1:n+1));
        shading interp;
        retazec=sprintf('Time=%g s',t);
        title(retazec);
        colorbar;
%       pause;

       TT(:,:,nn/step)=T;
end
```