NUMERICAL ANALYSIS PROJECT #1

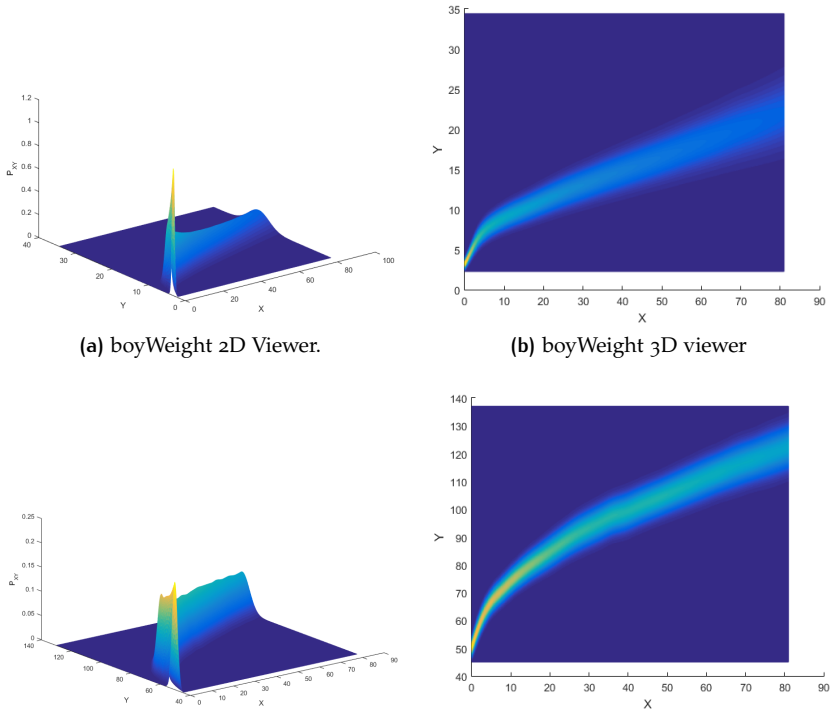
XINGLU WANG
Student Number: 3140102282
College of Information Science & Electronic Engineering

CONTENTS

LIST OF FIGURES

ABSTRACT

This project is about inner estimation based on Approximation theory(LSE) and natural Cubic Spline interpolation(NCSAPE).
First, I transform normal distribution into linear form. Then use LSE to get an relatively accurate μ and σ for norm distribution.
Then, I use NCSAPE to get continuous growth curve for μ and σ. In this



(a) boyWeight 2D Viewer.

(b) boyWeight 3D viewer

(c) boyHeight 2D viewer

(d) boyHeight 3D viewer

Figure 1: A number of pictures for overview.

way, we can know the standard height and weight for any month age child. Meanwhile I consider a GUI with appropriate interface. Design an application, interacting with user.

* *College of Information Science & Electronic Engineering, ZheJiang University, China*

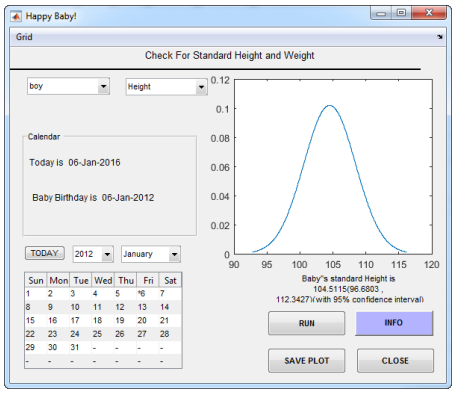


Figure 2: An overview of GUI

1 INTRODUCTION

This project is aimed to estimate the standard height and weight of baby and child. Apparently, there many factors that effect the height and weight, so it is difficult to get accurate value. The numerical method can give us deep insight into the complex statistics and provided us with an estimation whose error is acceptable.

2 INTERFACE

To design an complex system, I first consider its components' interface, which consists of its input and output data flow.

1. A suitable GUI, for user and application.
2. The relationship between each function model.

2.1 GUI interface

First, I design an GUI shown in Figure 2. It is event-drived. Using pop-upbox to choose boy, girl, weight or height and an convenient calendar to input the birthday of baby.
What will be resulted in if user gives an bad INPUT? It is shown in Figure 3b on the next page.

2.2 function interface

1. `function [year,month,day] = calendarGUI(¬,index,action,input)` When input is 'get', it will take time data from calendar on GUI.
2. `function [mupp,sigpp] = myPre(age,opt)` is the kernel function containing interpolation and approximation algorithm to estimate for any-aged baby.

The relationship between internal function can be complex, there many callback funcion. But basic and main function is calendarGUI(containing `run_wave_Callback` and `MainGUI_OpeningFcn`) and myPre(Containing my-Interp). The relation can be shown use some tool, but I find it still not very distinct. Reference to Figure 4 on the following page

(a) So young a baby **(b)** So old a baby

Figure 3: A number of pictures with no common theme.

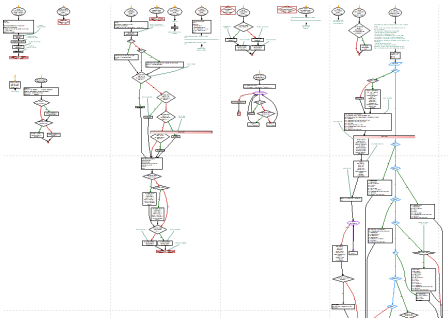


Figure 4: the complex function relation

3 IMPLEMENTATION

3.1 Overview

The overview of the complex data is shown on the beginning. Reference to figure 1 on page 2
First, we can see the data seams very similar to Normal Distribution. We can assume that the sample is mass enough ,therefore the distribution can be viewed as Normal Distribution. So I use the least square method via some identical transformation to get μ and σ as estimator for each discrete age's people.
Then, we observe from the plot that baby grow more fast than child. Since we do not know the explicit grew function, for accuracy we'd better use interpolation. And I choose cubic spline method.
The algorithm flow chart is in Figure 5 on the following page.

3.2 Approximation theory(LSE)

3.2.1 *transform the problem*

First, we can use identical transformation to convert the Normal function into a linear function. We construct $Z = \sqrt{\ln f + \frac{1}{2}\ln 2\pi + \ln \sigma}$, and submit f with Z

$$f(x \mid \mu, \sigma) = \frac{1}{\sigma\sqrt{2\pi}}\, e^{-\frac{(x-\mu)^2}{2\sigma^2}} \tag{1}$$

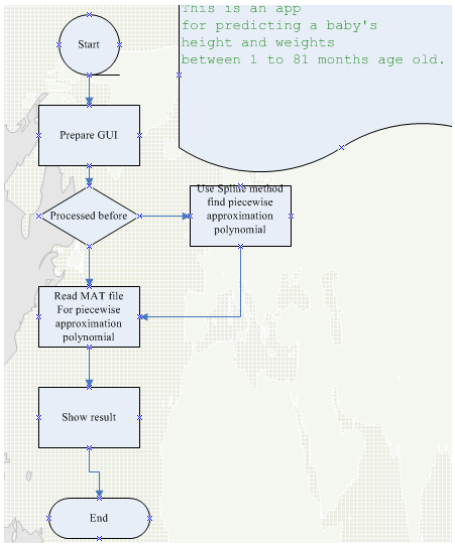


Figure 5: flow chart for algorithm

we get

$$Z = \frac{x}{\sqrt{2\sigma}} - \frac{u}{\sqrt{2\sigma}} \tag{2}$$

Then, how can we get discrete accurate data form the table shown below?

	-3SD	-2SD	-1SD	mean	+1SD	+2SD	+3SD
Height	44.7	46.4	48	49.7	51.4	53.2	55

We can know the standard normal possibility distribution function (norm-PDF), then we can get standard possibility dense at $X = \sigma, X = 2\sigma$ and so on.

```
mu = 0;
sigma = 1;
x = [-3,-2,-1,0,1,2,3];
standard = normpdf(x,mu,sigma);
```

Then we get table shown below:

	-3SD	-2SD	-1SD	mean	+1SD	+2SD	+3SD
Height	44.7	46.4	48	49.7	51.4	53.2	55
probability dense	0.0044	0.0540	0.2420	0.3989	0.2420	0.0540	0.0044

3.2.2 *Linear Least Square*

Definition 1 (LSE). Consider an overdetermined system $\sum_{j=1}^{n} X_{ij}\beta_j = y_i$, $(i = 1, 2, \ldots, m)$, of m linear equations in n unknown coefficients, $\beta_1, \beta_2, ..., \beta_n$, with $m > n$. This can be written in matrix form as

$$\mathbf{X\beta} = \mathbf{y}$$

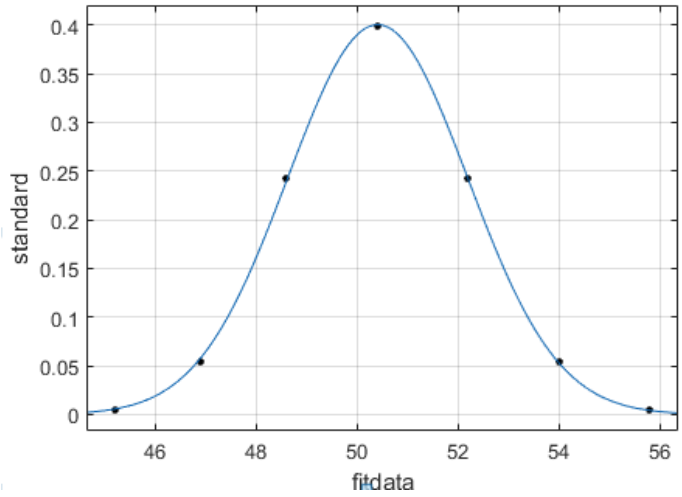


Figure 6: use LSE to fit normPDF

where

$$\mathbf{X} = \begin{bmatrix} X_{11} & X_{12} & \cdots & X_{1n} \\ X_{21} & X_{22} & \cdots & X_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ X_{m1} & X_{m2} & \cdots & X_{mn} \end{bmatrix}, \quad \boldsymbol{\beta} = \begin{bmatrix} \beta_1 \\ \beta_2 \\ \vdots \\ \beta_n \end{bmatrix}, \quad \mathbf{y} = \begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_m \end{bmatrix}.$$

Then, the LSE method is aim at getting an appropriate β.

$$\hat{\boldsymbol{\beta}} = \arg\min_{\boldsymbol{\beta}} S(\boldsymbol{\beta}),$$

where the objective function S is given by

$$S(\boldsymbol{\beta}) = \sum_{i=1}^{m} \left| y_i - \sum_{j=1}^{n} X_{ij}\beta_j \right|^2 = \left\| \mathbf{y} - \mathbf{X}\boldsymbol{\beta} \right\|^2.$$

3.2.3 *Fit Result*

To illustrate the fit result, we use boy height at 1 month age old. Reference to Figure 6. We can see the data is not all pass the fit curve(magnify, and we will see it!), which prove this step is required if we are pursuing for accuracy.
But since the original data is from mass sample, the revision of σ and μ is not very remarkable. If we want to implement this application on mobile. We can simply use adjacent data to minus.

3.3 natural Cubic Spline interpolation(NCSAPE)

Definition 2 (CSAPE)**.** The cubic spline is given by the function values in the nodes and derivative values on the edges of the interpolation interval (either of the first or second derivatives)

I choose natural CSAPE.

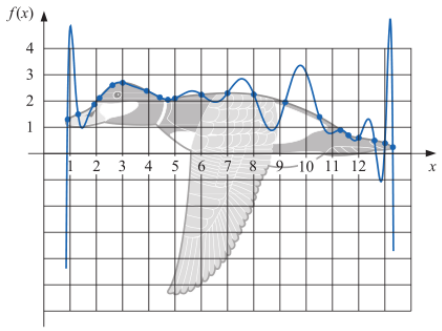


Figure 7: illustration of Runge's phenomenon

3.3.1 *Why I choose CSAPE?*

The most important reason is that the interpolation error can be made small with 3 degree polynomials for the spline. As a law of natural, children must grow step by step, become taller and stronger gradually. They cannot become very tall suddenly! And CSAPE meets the condition.
Cubic Spline interpolation avoids the problem of Runge's phenomenon, in which oscillation can occur between points when interpolating using high degree polynomials. And the Runge's phenomenon is illustrated in 7

3.3.2 *Why I choose **natural** CSAPE?*

Let us compare natural CSAPE with clamp CSAPE:

1. If the exact values of the first derivative in both boundaries are known, such spline is called clamped spline, or spline with exact boundary conditions. This spline has interpolation error $O(h^4)$.

2. If the value of the first (or second) derivative is unknown, we can set the so-called natural boundary conditions $S''(A) = 0$, $S''(B) = 0$. Thus, we get a natural spline. The natural spline has interpolation error $O(h^4)$ in the inner nodes. The closer to the boundary nodes the more the error becomes. In the inner nodes the interpolation accuracy is much better.

The reason lies in that we do not know the accurate border message about growth curve, and we won't inquiry standard data for a unborn baby(too young) and a adult(too old), which means the bound data will in the least be cared for.

3.3.3 *Construct Cubic Spline Interpolate:*

As shown below, The construction of the cubic spline is based on the belief that the function value ,derivation and second derivation of the interplant function agree with each other at the nodes.

Given a function f defined on $[a,b]$ and a set of nodes $a = x_0 < x_1 < \cdots < x_n = b$, a **cubic spline interpolant** S for f is a function that satisfies the following conditions:

(a) $S(x)$ is a cubic polynomial, denoted $S_j(x)$, on the subinterval $[x_j, x_{j+1}]$ for each $j = 0, 1, \ldots, n-1$;

(b) $S_j(x_j) = f(x_j)$ and $S_j(x_{j+1}) = f(x_{j+1})$ for each $j = 0, 1, \ldots, n-1$;

(c) $S_{j+1}(x_{j+1}) = S_j(x_{j+1})$ for each $j = 0, 1, \ldots, n-2$; (*Implied by* **(b)**.)

(d) $S'_{j+1}(x_{j+1}) = S'_j(x_{j+1})$ for each $j = 0, 1, \ldots, n-2$;

(e) $S''_{j+1}(x_{j+1}) = S''_j(x_{j+1})$ for each $j = 0, 1, \ldots, n-2$;

(f) One of the following sets of boundary conditions is satisfied:

(i) $S''(x_0) = S''(x_n) = 0$ (**natural** *(or free)* **boundary**);

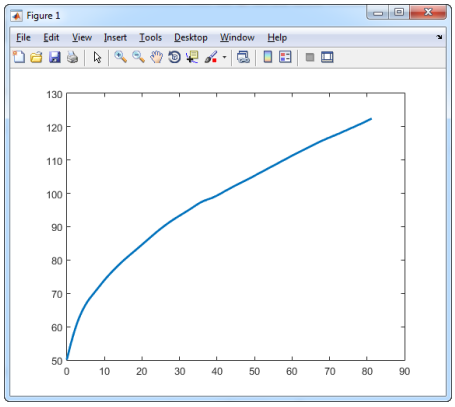
(ii) $S'(x_0) = f'(x_0)$ and $S'(x_n) = f'(x_n)$ (**clamped boundary**). ■

Here is cubic spline implement by myself:

```
function pp = myInterp(x,y)
%pp=csape(x',y');
pp = [];
n = length(x);
%% make sure comlumn
[nr, nc] = size(y);
if nr == 1
    y = reshape(y, nc, 1);
    nr = nc;
end
[nr, nc] = size(x);
if nr == 1
    x = reshape(x, nc, 1);
    nr = nc;
end
%%
if size(x) ≠ size(y)
    error('x and y are of different size');
end
dx = [0; diff(x); 0];
dxx = dx(1:n) + dx(2:n+1);
% d_xx_j = h_j+h_{j+1}
%%
M = spdiags([[dx(2:n)./dxx(2:n); 0] 2*ones(n,1) [0; ...
    dx(2:n)./dxx(1:n-1)]], -1:1, n,n);
dy_l = 0;
dy_r = 0;

b = diff(y) ./ dx(2:n);
c = 6 * diff([dy_l; b; dy_r])./ dxx;

%% For natural spline interpolation
c(1)     = 0;
c(n)     = 0;
M(1,2)   = 0;
M(n,n-1) = 0;
c = M\c;
d = diff(c)./dx(2:n);
b = b  - dx(2:n).* (c(1:n-1)/3 + c(2:n)/6);

%% now compute the values yy
xx=x;
yy = zeros(size(xx));
for i=1:nr-1
    I = find(xx ⩽ x(i+1) & xx ⩾ x(i));
    yy(I) = y(i) + b(i)*(xx(I)-x(i))+c(i)/2*(xx(I)-x(i)).^2 + ...
        d(i)/6*(xx(I)-x(i)).^3;
end
pp=csape(xx,yy);
```

```
% c=c(1:end-1);
% y=y(1:end-1);
%% make piecewise poly
%pp=mkpp(x',[d,b,c,y]);
pp = csape(xx,yy);
% Here use csape is only make piecewise ploynomial!
% Becuase my [a,b,c,d] is different from pp-form in matlab. ...
    But I want to
% use its plot function. To use its fit-function is just for ...
    convenience!
```

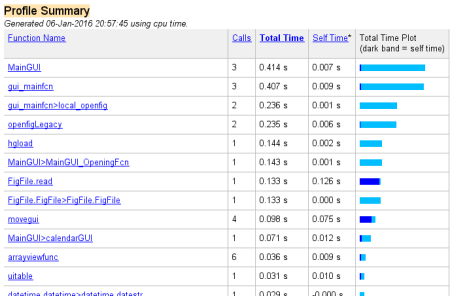
And the result for boy height CSAPE is shown below:



3.3.4 *Uniqueness of natural CSAPE*

Theorem 1. *If* f *is defined at* $a = x_0 < x_1 < ... < x_n = b$,*then* f *has a unique natural spline interpolant S; that is, a spline interpolant that satisfies the natural boundary is unique.*

Proof. Use the continuity and derivation condition, we can get an linear equation system that specifies our CSAPE.
The matrix A is strictly diagonally dominant, that is, in each row the magnitude of the diagonal entry exceeds the sum of the magnitudes of all the other entries in the row.A linear system with a matrix of this form have a unique solution.



Profile Summary

Generated 06-Jan-2016 20:57:45 using cpu time.

Function Name	Calls	Total Time	Self Time*	Total Time Plot (dark band = self time)
MainGUI	3	0.414 s	0.007 s	
gui_mainfcn	3	0.407 s	0.009 s	
gui_mainfcn>local_openfig	2	0.236 s	0.001 s	
openfigLegacy	2	0.235 s	0.006 s	
hgload	1	0.144 s	0.002 s	
MainGUI>MainGUI_OpeningFcn	1	0.143 s	0.001 s	
FigFile.read	1	0.133 s	0.126 s	
FigFile.FigFile>FigFile.FigFile	1	0.133 s	0.000 s	
movegui	4	0.098 s	0.075 s	
MainGUI>calendarGUI	1	0.071 s	0.012 s	
arrayviewfunc	6	0.036 s	0.009 s	
uitable	1	0.031 s	0.010 s	
datetime.datetime>datetime.datestr	1	0.029 s	0.000 s	

Figure 8: the complexity

Proof The boundary conditions in this case imply that $c_n = S''(x_n)/2 = 0$ and that

$$0 = S''(x_0) = 2c_0 + 6d_0(x_0 - x_0),$$

so $c_0 = 0$. The two equations $c_0 = 0$ and $c_n = 0$ together with the equations in (3.21) produce a linear system described by the vector equation $A\mathbf{x} = \mathbf{b}$, where A is the $(n+1) \times (n+1)$ matrix

$$A = \begin{bmatrix} 1 & 0 & 0 & \cdots & 0 \\ h_0 & 2(h_0+h_1) & h_1 & & \\ 0 & h_1 & 2(h_1+h_2) & h_2 & \\ \vdots & & \ddots & \ddots & 0 \\ & & h_{n-2} & 2(h_{n-2}+h_{n-1}) & h_{n-1} \\ 0 & \cdots & 0 & 0 & 1 \end{bmatrix},$$

and $\mathbf{b}$ and $\mathbf{x}$ are the vectors

$$\mathbf{b} = \begin{bmatrix} 0 \\ \frac{3}{h_1}(a_2 - a_1) - \frac{3}{h_0}(a_1 - a_0) \\ \vdots \\ \frac{3}{h_{n-1}}(a_n - a_{n-1}) - \frac{3}{h_{n-2}}(a_{n-1} - a_{n-2}) \\ 0 \end{bmatrix} \quad \text{and} \quad \mathbf{x} = \begin{bmatrix} c_0 \\ c_1 \\ \vdots \\ c_n \end{bmatrix}.$$

□

4 DISCUSSION

4.1 Complexity analysis

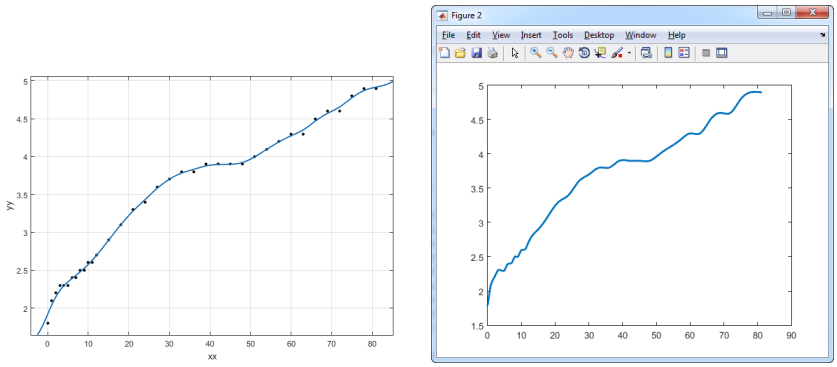
The total computational complexity is $O(C^2N)$. Reference to the analysis of Matlab [8]

4.1.1 *For LSE*

Refer to normal equation — $\beta = (X^TX)^{-1}X^Ty$, we can conclude that it costs $O(C^2N)$

4.1.2 *For CSAPE*

We can solve the well-posed linear system with iteration method, so it cost $O(N)$.



(a) smoothing spline **(b)** cubic spline

Figure 9: CSAPE V.S. SSAPE

4.2 Improvement(SSAPE)

As we believe, children grow gradually, which means the curve of growth must be as smooth as possible. So there are more suitable method to approach the curve —- Smoothing Spline(SSAPE).
Understand the notion visual oriented, we can refer to Figure 9a

SSAPE adds an roughness penalty function to the objective function and use the method of optimization to avoid affection of noisy observations.

Definition 3 (Smoothing Spline)**.** Let $(x_i, Y_i); x_1 < x_2 < \cdots < x_n, i \in \mathbb{Z}$ be a sequence of observations, modeled by the relation $Y_i = \mu(x_i)$. The smoothing spline estimate $\hat{\mu}$ of the function μ is defined to be the minimizer (over the class of twice differentiable functions) of

$$\sum_{i=1}^{n} (Y_i - \hat{\mu}(x_i))^2 + \lambda \int_{x_1}^{x_n} \hat{\mu}''(x)^2 \, dx \tag{3}$$

And we can use the matlab function to achieve our idea.

```
function fitresult=SmoothingSpline(X,Y)
%% Using smoothing spline.
ft = fittype( 'smoothingspline' );
opts = fitoptions( 'Method', 'SmoothingSpline' );
opts.SmoothingParam = 0.115222516467702;
[fitresult, ¬] = fit( X', Y', ft, opts );
figure;
plot( fitresult, X', Y' );
```

APPENDIX

Attach the MainGUI function:

```
%%
%Hello!
%This is an app for predicting a baby's height and weights ...
    between 1 to 81
% months age old.

```

```
function varargout = MainGUI(varargin)

gui_Singleton = 1;
gui_State = struct('gui_Name',       mfilename, ...
    'gui_Singleton',  gui_Singleton, ...
    'gui_OpeningFcn', @MainGUI_OpeningFcn, ...
    'gui_OutputFcn',  @MainGUI_OutputFcn, ...
    'gui_LayoutFcn',  [] , ...
    'gui_Callback',   []);
if nargin & isstr(varargin{1})
    gui_State.gui_Callback = str2func(varargin{1});
end
if nargout
    [varargout{1:nargout}] = gui_mainfcn(gui_State, varargin{:});
else
    gui_mainfcn(gui_State, varargin{:});
end


%% ...
    ------------------------------------------------------------------------
function MainGUI_OpeningFcn(hObject, eventdata, handles, varargin)
global Now
format long; clc;
[yeN,moN,daN]=calendarGUI(0,0,'create',gcf);
Now=datetime([yeN,moN,daN]);
set(handles.text25,'String',[get(handles.text25,'String'), '  ...
    ' , datestr(Now)]);
handles.output = hObject;
movegui(hObject,'onscreen')              % To display ...
    application onscreen
movegui(hObject,'center')                % To display ...
    application in the center of screen
set(handles.gridopt,'checked','off')     % To check the grid ...
    option
handles.Leg = [];                        % Initialize Leg ...
    variable for later use
guidata(hObject, handles);

%% ...
    ------------------------------------------------------------------------
function varargout = MainGUI_OutputFcn(hObject, eventdata, ...
    handles)
varargout{1} = handles.output;

%% ...
    ------------------------------------------------------------------------
function run_wave_Callback(hObject, eventdata, handles)
global Now
[yeB,moB,daB] = calendarGUI(0,0,'get');
Bir=datetime([yeB,moB,daB]);
set(handles.text26,'String',[ 'Baby Birthday is  ' , ...
    datestr(Bir)]);
% formatIn = 'mm/dd/yyyy';
% datenum(DateString,formatIn)
age=(datenum(Now)-datenum(Bir))/30;
if age<0
     msgbox('Are You Sure Your Baby has been born QAQ ?', ...
            'Warning','None','modal')
        return
elseif age>100
    msgbox('Month age is suggested ⩽81, otherwise the result ...
        maybe inaccurate! ', ...
            'Warning','None','modal')
        return
end

```

```
if get(handles.boyOr,'value') == 1 && ...
    get(handles.heightOr,'value') == 1     % loop for B-Splines
   opt='boyHeight';

elseif get(handles.boyOr,'Value') == 1 && ...
    get(handles.heightOr,'Value')==2      % loop for Cardinal ...
    Spline
   opt='boyWeight';

elseif get(handles.boyOr,'Value') == 2 && ...
    get(handles.heightOr,'Value') ==  1    % loop for Naive ...
    Basis Function
   opt='girlHeight';

elseif get(handles.boyOr , 'Value' ) ==2 && ...
    get(handles.heightOr,'Value') == 2
   opt='girlWeight';
end
[mupp,sigpp]=myPre(age,opt);
mu=ppval(mupp,age);
sig=ppval(sigpp,age);
xx=linspace(mu-3*sig,mu+3*sig,500);
yy=normpdf(xx,mu,sig);
plot(xx,yy);
res=mu;
if strfind(opt,'Height')
    set(handles.text28,'String',['Baby"s standard Height is ...
        ',num2str(res),'(',num2str(res-2*sig),' , ...
        ',num2str(res+2*sig),')','(with 95% confidence ...
        interval)']);
elseif strfind(opt,'Weight')
    set(handles.text28,'String',['baby"s standard Weight is ...
        ',num2str(res),'(',num2str(res-2*sig),' , ...
        ',num2str(res+2*sig),')','(with 95% confidence ...
        interval)']);
end
if strcmp(get(handles.gridopt,'checked'),'off')
    set(handles.display_plot,'XGrid','Off','YGrid','Off','ZGrid','Off') ...
        % Make the grid invisible
else
    set(handles.display_plot,'XGrid','On','YGrid','On','ZGrid','On') ...
          % Make the grid visible
end
%handles.Leg = LegendVar;
guidata(hObject, handles);

%% ...
    ------------------------------------------------------------------------
function info_Callback(hObject, eventdata, handles)
helpwin('MainGUI.m')

%% ...
    ------------------------------------------------------------------------
function close_button_Callback(hObject, eventdata, handles)
close(gcbf) % to close GUI

%% ...
    ------------------------------------------------------------------------
% Executes on button press in save_plot.
function save_plot_Callback(hObject, eventdata, handles)

h = get(gcf,'CurrentAxes');
figure(1);
copyobj(h,gcf);
set(gca,'FontSize',12);
c = copyobj(handles.Leg,gcf);
set(gcf, 'PaperPosition', [2 1 8 4]);
```

```
112 print( gcf, '-dpng', 'plot.png' );
113 close(1);
114
115 % ... ------------------------------------------------------------------------
116 % function i_CreateFcn(hObject, eventdata, handles)
117 %
118 % %% ... ------------------------------------------------------------------------
119 % function i_Callback(hObject, eventdata, handles)
120 %
121 % %% ... ------------------------------------------------------------------------
122 % function k_CreateFcn(hObject, eventdata, handles)
123 %
124 % ... ------------------------------------------------------------------------
125 % function k_Callback(hObject, eventdata, handles)
126
127 %% ... ------------------------------------------------------------------------
128 function Gridmenu_Callback(hObject, eventdata, handles)
129
130 %% ... ------------------------------------------------------------------------
131 function grid_onoff_Callback(hObject, eventdata, handles)
132 if strcmp(get(handles.gridopt,'checked'),'on')
133     set(handles.gridopt,'checked','off')           % To ...
        uncheck the grid option
134     set(handles.display_plot,'XGrid','Off','YGrid','Off','ZGrid','Off') ...
        % Make the grid invisible
135 else
136     set(handles.gridopt,'checked','on')            % To check ...
        the grid option
137     set(handles.display_plot,'XGrid','On','YGrid','On','ZGrid','On') ...
          % Make the grid visible
138 end
139
140 %% ... ------------------------------------------------------------------------
141 function boyOr_CreateFcn(hObject, eventdata, handles)
142
143 %% ... ------------------------------------------------------------------------
144 function boyOr_Callback(hObject, eventdata, handles)
145
146 %------------------------------------------------------------------------
147 % function n_CreateFcn(hObject, eventdata, handles)
148 %
149 % % ... ------------------------------------------------------------------------
150 % function n_Callback(hObject, eventdata, handles)
151
152 %% --- Executes on selection change in heightOr.
153 function heightOr_Callback(hObject, eventdata, handles)
154
155 %% --- Executes during object creation, after setting all ...
    properties.
156 function heightOr_CreateFcn(hObject, eventdata, handles)
157 if ispc && isequal(get(hObject,'BackgroundColor'), ...
    get(0,'defaultUicontrolBackgroundColor'))
158     set(hObject,'BackgroundColor','white');
159 end
160
161 %%
162 function [year,month,day] = calendarGUI(¬,index,action,input)
```

```
% Simple calendar GUI which can be incorporated into a figure, a uipanel or
% a uitab.
% In order to use it into a different figure, uipanel or uitab just set
% input as the handle like :
% [year,month,day] = calendarGUI(0,0,'create',handle)
% to only get values use the same function calling :
% [year,month,day] = calendarGUI(0,0,'get')
% the 4 parameters in the beginning can be set to other values :
% the start_year is the first year in the Year popupmenu box
% the year_offset is the number of year after the current year
% xpos and ypos are the position of the calendar from left corner of the
% handle element
% It also can be used simply by calling :
% [year,month,day] = calendarGUI(0,0,'create')
% and will output the year, month and day when closing the figure.
% Using it this way does not allow to use the 'get' action functionality


start_year = 2005;
year_offset = 5;
xpos = 20;
ypos = 20;

switch action
    case 'create'
        if nargin < 4
            parent = figure('Position',[750   520   xpos+215   ypos+170],...
                'MenuBar','none','Name','** Calendar **',...
                'DockControls','off','Toolbar','none','NumberTitle','off',...
                'CloseRequestFcn',{@calendarGUI,'fcnClose'},...
                'Resize','off','Tag','SelfRunning');
        else
            parent = input;
        end

        % Create the calendar
        [year,month,day] = datevec(now());
        ht = uitable('Parent',parent,...
            'TooltipString','Calendar Date',...
            'Tag','Day','ColumnEditable',false,...
            'CellSelectionCallback',...
            {@calendarGUI,'fcnDay'});
        set(ht,'RowName',[])
        set(ht,'ColumnName',{'Sun' 'Mon' 'Tue' 'Wed' 'Thu' 'Fri' 'Sat'})
        set(ht,'ColumnWidth',{30 30 30 30 30 30 30})
        set(ht,'Units','Pixels');
        set(ht,'Position',[xpos ypos 211.1 130])
        date = produce_date_array(year,month,day);
        set(ht,'Data',date);
        set(ht,'UserData',[year,month,day]);

        % create Month box
        uicontrol('Parent',parent,'Style','popupmenu',...
            'Position',[xpos+130 ypos+145 80 20],...
            'Callback',{@calendarGUI,'fcnMonth'},...
            'Tag','Month',...
            'String',{'January' 'February' 'March' 'April' 'May'...
            'June' 'July' 'August' 'September' 'October' 'November'...
            'December'},...
```

```
222             'Value',month);
223
224         % create Today pushbutton
225         htoday = ...
                uicontrol('Parent',parent,'Style','pushbutton',...
226             'Position',[xpos ypos+145 55 20],...
227             'Callback',{@calendarGUI,'fcnToday'},...
228             'Tag','Today','String','TODAY');
229
230         % Create Year box
231         y_vector = start_year:year + year_offset;
232         y = {};
233         for ii = 1:length(y_vector)
234             y = [y {mat2str(y_vector(ii))}];
235         end
236         uicontrol('Parent',parent,'Style','popupmenu',...
237             'Position',[xpos+65 ypos+145 55 20],...
238             'Callback',{@calendarGUI,'fcnYear'},...
239             'Tag','Year',...
240             'String', y,...
241             'Value',year-(start_year-1));
242
243         if strcmp(get(parent,'Tag'),'SelfRunning')
244             waitfor(htoday)
245             [year,month,day] = calendarGUI(0,0,'get');
246             delete(parent)
247         end
248
249
250     case 'fcnYear'
251         hy = findobj('Tag','Year');
252         ht = findobj('Tag','Day');
253         ymd = get(ht,'UserData');
254         month = ymd(2);
255         day = ymd(3);
256         year = get(hy,'Value');
257         year = year + (start_year-1);
258         date = produce_date_array(year,month,day);
259         set(ht,'Data',date);
260         set(ht,'UserData',[year month day]);
261
262     case 'fcnMonth'
263         hm = findobj('Tag','Month');
264         ht = findobj('Tag','Day');
265         ymd = get(ht,'UserData');
266         year = ymd(1);
267         day = ymd(3);
268         month = get(hm,'Value');
269         date = produce_date_array(year,month,day);
270         set(ht,'Data',date);
271         set(ht,'UserData',[year month day]);
272
273     case 'fcnToday'
274         [year,month,day] = datevec(now());
275         date = produce_date_array(year,month,day);
276         ht = findobj('Tag','Day');
277         hm = findobj('Tag','Month');
278         set(hm,'Value',month);
279         hy = findobj('Tag','Year');
280         set(hy,'Value',year-(start_year-1));
281         set(ht,'Data',date);
282         set(ht,'UserData',[year month day]);
283
284     case 'fcnDay'
285         if ~isempty(index.Indices) % because the function runs ...
                2 times
286             xi = index.Indices(1);
```

```
            yi = index.Indices(2);
            ht = findobj('Tag','Day');
            data = get(ht,'Data');
            get(ht,'UserData');
            ymd = get(ht,'UserData');
            year = ymd(1);
            month = ymd(2);
            day = data(xi,yi);
            day = strrep(day,'*','');
            day = str2double(day);
            date = produce_date_array(year,month,day);
            set(ht,'Data',date);
            set(ht,'UserData',[year month day]);
        end

    case 'get'
        ht = findobj('Tag','Day');
        ymd = get(ht,'UserData');
        %ymd=ymd(end);
        %ymd=ymd{:};
        year = ymd(1);
        month = ymd(2);
        day = ymd(3);

    case 'fcnSetDate'
        if size(input,2) == 3
            year = input(1);
            month = input(2);
            day = input(3);
            date = produce_date_array(year,month,day);
            ht = findobj('Tag','Day');
            hm = findobj('Tag','Month');
            set(hm,'Value',month);
            hy = findobj('Tag','Year');
            set(hy,'Value',year-(start_year-1));
            set(ht,'Data',date);
            set(ht,'UserData',[year month day]);
        else
            msgbox('Input must be a 1x3 vector [year month day]')
        end

    case 'fcnClose'
        htoday = findobj('Tag','Today');
        delete(htoday);
end

%% _____________________________________
% function to get date
function [date] = produce_date_array(year,month,day)

date = calendar(year,month);
date = mat2cell(date(:),ones(size(date,1)*size(date,2),1));
for i = 1:length(date)
    if date{i} == 0
        date{i} = '-';
    elseif date{i} == day
        date{i} = sprintf('*%d',date{i});
    else
        date{i} = mat2str(date{i});
    end
end
date = reshape(date,6,7);
```