Introduction

Consider two 2D surfaces, whose plots via `surf` are shown in Figure 1 below:

![Surface plots of 2 functions](image)

(a) Surface 1  (b) Surface 2

Figure 1: Surface plots of 2 functions. Surface 1 is the peaks function $f_1(x, y) = (3(1-x)^2 \times exp(-(x^2) - (y+1)^2) \times 10(x/5 - x^3 - y^5) \times exp(-x^2 - y^2))^{1/3} \times exp(-(x+1)^2 - y^2))$ and Surface 2 is the 2D Gaussian function $f_2(x, y, Zscale, sigma) = Zscale \times exp(-(x \times x + y \times y)/(2 \times sigma^2))$. Both graphs show the numerical and symbolic evaluated integral values printed on them.

The first surface is given by the equation of the peaks function as:

$$f_1(x, y) = z = 3(1-x)^2 \times exp(-(x^2) - (y+1)^2) - 10(x/5 - x^3 - y^5) \times exp(-x^2 - y^2) - 1/3 \times exp(-(x+1)^2 - y^2)$$

(1)

while a simple, second function, 2D Gaussian, is specified as:

$$f_2(x, y, Zscale, sigma) = Zscale \times exp(-(x \times x + y \times y)/(2 \times sigma^2)),$$

(2)
where we use \( Z_{scale} = 8.0 \) and \( \sigma = 2.0 \) for this assignment. You have already done an animation in Lab 5.

This assignment requires you to write a MatLab program that performs an animation of these 2 function surface functions, warping the first surface into the second surface, then the second surface back into the first surface.

**Plotting the Functions**

The various tasks in this assignment include:

1. First, you have to plot these the 2 functions. Use \( x_{min} \) and \( y_{min} \) values of -4 and \( x_{max} \) and \( y_{max} \) values of +4, along with \( \Delta x = \Delta y = 0.05 \), to generate \( X \) and \( Y \) arrays from `meshgrid`. The \( z \) values are computed by MatLab (but override these values with \( z_{min} = -8 \) and \( z_{max} = +8 \) using `axis` for all the figures you display in this assignment). Then the 3D depth values, \( Z_1 \) and \( Z_2 \), can be generated by a vectorized calculation for these 2 functions. \( Z_1 \) is the vectorized form of \( f_1(x, y) \) while \( Z_2 \) is the vectorized form of \( f_2(x, y) \). Using `meshgrid(x, y)`, where \( x \) is \( x_{min} : \Delta x : x_{max} \) and \( y \) is \( y_{min} : \Delta y : y_{max} \), compute \( X \) and \( Y \) and plot the 2 functions, given by \((X, Y, Z_1)\) and \((X, Y, Z_2)\).

2. Next, perform numerical and symbolic integration on these 2 functions and use `text` to print out these values in blue and red respectively. Choose an appropriate fontsize and appropriate 3D coordinates for `text` to positioning the text while print the numerical and symbolic integration results on the graphs.

   (a) To numerically integrate the 2 functions you need to write an anonymous function defined for two surfaces as:

   ```matlab
   fun1 = @(X,Y) (your vectorized expression for f1(x,y));
   num_area1=integral2(fun1,xmin,xmax,ymin,ymax);
   fun2 = @(X,Y) (your vectorized expression for f2(x,y));
   num_area2=integral2(fun2,xmin,xmax,ymin,ymax);
   ```

   Here `fun1` and `fun2` are the “handles” (or pointers) to **anonymous** functions. You can pass this handle to a function as a parameter to another function. Effectively, you can have a function as a parameter to another function. In this
assignment, you can integrate functions, fun1 and fun2, using integral2. MatLab function, integral2, evaluates the area under this function using numerical quadrature.

(b) To symbolically integrate the 2 functions you need to declare \( f_1, f_2, x \) and \( y \) to be symbol variables, compute \( f_1 \) and \( f_2 \) and then evaluate their symbolic integral values.

\[
\% \text{ declare } f, x \text{ and } y \text{ as MatLab symbols} \\
syms f1 f2 x y
\]

and then use:

\[
f1=... \\
sym\_area1=eval(int(int(f1,y,ymin,ymax),x,xmin,xmax));
\]

\[
f2=... \\
sym\_area2=eval(int(int(f2,y,ymin,ymax),x,xmin,xmax));
\]

to integrate the 2 surfaces. \( f_1 \) and \( f_2 \) are each specified over 2 lines here for printing purposes, you should specify them on a single line in your program. Note the use of eval to evaluate the symbolic integration result (usually a complex expression that could be evaluated using a calculator). For this assignment numerical and symbolic integration always yield the same answers.

3. You can use the title command to print out the mathematical formula for each of the 2 surfaces. [An expression is interpreted as latex in MatLab, with \(_\) indicating a subscript and a ^ \ indicating a superscript. Thus \( x_2 \) is \( x_2 \) while \( x^2 \) is \( x^2 \).] Also use xlabel, ylabel and zlabel to label the \( x, y \) and \( z \) axes in your figures. Color these using black pr blue, green and red (as indicated by the appropriate figures) and set the fontsize to a large value. Use the 3D version of text to print out the numerical and symbolic integration results on these graphs (see below). Lastly, print out the Gaussian parameters, Zvalue and sigma used by your program. You have to chose the \( x, y \) and \( z \) values to position these integration values on the plots.

4. You create the animation by warping \( Z_1 \) into \( Z_2 \), then \( Z_2 \) back into \( Z_1 \). So the initial and final surfaces are the same. To code the warping for \( Z_1 \) to \( Z_2 \) you can use something like:
for t=0:delta_t:1
    Z=Z1*(1-t)+Z2*(t);
    surf(X,Y,Z)
    axis([xmin xmax ymin ymax zmin zmax]);
    ...
    pause(display_pause);
    drawnow;
end % for t

The statement $Z=Z1*(1-t)+Z2*(t)$ does the surface warping, i.e. this is the linear interpolation of the two surfaces $Z1$ and $Z2$. For $t = 0$, $Z=Z1$ while for $t = 1$, $Z=Z2$. Intermediate values of $t$ give you the various combined surfaces of $Z1$ and $Z2$. $\delta t$ is a small number, say 0.05. Thus, when this runs you will see a total of 21 surfaces displayed rapidly as $Z1$ warps into $Z2$. If the display is too rapid, you can pause a small amount of time, display_pause, between adjacent displays to slow things down. After this loop executes, a second loop is needed to warp $Z2$ back into $Z1$. Use the text command to print out the numerical and symbolic integration areas. Set variables xpos1, ypos1 and zpos1 and xpos2, ypos2 and zpos2 appropriately, where zpos1 and zpos2 are the vertical dimension for this figure. Trial and error is required here. You could use grid on and box on to get some good initial values. Note that the text is printed in magenta, the numerical and symbolic integration areas are linearly interpolated to correspond with the current surface being displayed. After warping $Z1$ into $Z2$, you need to warp $Z2$ into $Z1$. At this point you will have your animation working. Figure 2 show the peaks surface and the 2D Gaussian surface 2D Gaussian function, with the axes labelled, a title in orange and pink and the numerically and symbolically evaluates integral values in red. For orange and pink (which are not MatLab builtin colors, use \texttt{\textcolor{rgb}{1 0.6 0}} and \texttt{\textcolor{rgb}{1 0.4 0.7}} in the character strings in the title command. Figure 2 shows the peaks surface warping into the 2D Gaussian at $t$ values of 0.0, 0.35, 0.75 and 1.0.

5. Finally, the last task is to make the animation figure bigger than normal and to custom fit your computer’s screen (your program will run on all screen sizes with relative screen sizing done automatically). A function \texttt{set_screen_parameters()} is provided on the course webpage to do this. Call this before you start your animation but after you
Figure 2: The peaks and Gaussian functions, $f_1(x, y)$ and $f_2(x, y, Zscale, sigma)$, in animation for $t$ values of 0, 0.35, 0.7 and 1.0.

compute and display the first 2 figures. Have a statement asjing you to type <return> before you call *set_screen_parameters*.

You should write 3 MatLab functions, *ass3_2019* for your overall code and 2 functions for your *peaks* and 2D *Gaussian* functions. Do not use MatLab’s *peaks* function anywhere.