DOING PHYSICS WITH MATLAB

Ian Cooper

matlabvisualphysics@gmail.com

A COMPUTATIONAL APPROACH TO ELECTROMAGNETIC THEORY

CHAPTER 2

VECTOR CALCULUS

DOWNLOAD DIRECTORIES FOR MATLAB SCRIPTS

Google drive

GitHub

cemDiff01.m **Matlab functions** gradient del2

1st and 2nd derivatives of the function $y = \sin(k\,x)$

Numerical approximations for the derivatives

gradient and del2 functions

Comparison of numerical solutions with the exact answers

cemDiff02.m Matlab function **gradient**

gradient function f(x,y) ∇f

contourf plot of the function f(x,y)

quiver plot of the vector field ∇f

cemDiff03.m Matlab function **divergence**

divergence of a vector field $\nabla \bullet \vec{V}\left(V_x\ V_y\right)$

surf plot of $\nabla \bullet \vec{V}\left(V_x\ V_y\right)$

contourf and quiver plots of $\vec{V}$ and $\nabla \bullet \vec{V}\left(V_x\ V_y\right)$

cemDiff04.m Matlab functions **divergence curl**

divergence of a vector field $\nabla \bullet \vec{V}\left(V_x\ V_y\ V_z\right)$

curl of a vector field $\nabla \times \vec{V}\left(V_x\ V_y\ V_z\right)$

quiver3 plot of $\vec{V}$

slice plot of $\nabla \bullet \vec{V}\left(V_x\ V_y\ V_z\right)$

quiver3 plot of $\nabla \times \vec{V}\left(V_x\ V_y\ V_z\right)$

cemDiff05.m Matlab function **del2**

Laplacian $\nabla^2 f$ of the a scalar function f(x,y) surf plots

Laplacian $\nabla^2 \vec{V}$ of the vector function $\vec{V}\left(V_x\ V_y\ V_z\right)$ slice & quiver3 plots

mqm002.m

First and second derivatives performed as a **matrix** operation.

Topics

FIRST DERIVATIVE OF A ONE VARIABLE FUNCTION

Consider the one variable function $y(x) \equiv f(x)$. Then the **first derivative** of the function $y(x)$ is

$$(1) \qquad \frac{dy(x)}{dx} \equiv \frac{dy}{dx} = \lim_{\Delta x \to 0}\left(\frac{y(x+\Delta x)}{\Delta x}\right)$$

The first derivative tells us how rapidly the function $y(x)$ varies when we change the value of x by a tiny amount dx.

$$(2) \qquad dy = \left(\frac{dy}{dx}\right)dx$$

If we change x by the amount dx then the change in y is dy and is proportional to dx with the constant of proportionality equal to the first derivative dy/dx. Graphically the first derivative gives the **slope** or **gradient** of the tangent of the curve $y(x)$ verses x.

We can approximate the first derivative at x by the following:

$$(3a) \qquad \left.\frac{dy}{dx}\right|_x \approx \frac{y(x+\Delta x) - y(x)}{\Delta x} \qquad \text{forward approximation}$$

$$(3b) \qquad \left.\frac{dy}{dx}\right|_x \approx \frac{y(x) - y(x-\Delta x)}{\Delta x} \qquad \text{backward approximation}$$

$$(3c) \qquad \left.\frac{dy}{dx}\right|_x \approx \frac{y(x+\Delta x) - y(x-\Delta x)}{2\Delta x} \qquad \text{central difference approximation}$$

Mathematically, equations 3 are only correct in the limit $\Delta x \to 0$. In calculating derivatives numerically, Δx has to be small enough to provide sufficient accuracy of the result. There is an optimal value for Δx, since, if Δx is too small you get round-off errors. For most applications, the central difference approximation is preferred over the forward or backward methods. It is more difficult to achieve good accuracy in numerical differentiation compared to numerical integration. The main reason comes from the fact that we are taking the ratio of two differences.

In general, we do not have any difficulties in differentiating the functions normally encountered in physics, and as a result numerical methods to calculate derivatives were often not employed in the past. However, with the increasing use of computers and their greater speed and larger memory, numerical approaches in solving problems have become very important.

The Matlab mscript **cemDiff01.m** is used to find the derivative of the sine function

$$(4) \qquad y = \sin(k\,x) \quad k = \frac{2\,\pi}{\lambda} \qquad 0 \le x \le 100$$

where k (wave number) and λ (wavelength) are constants. The x values are given by N equally spaced grid points which are indexed from 1 to N. The spacing between the grid points is Δx. The larger the number of grid points, then the smaller the increment Δx.

The analytical (exact) function for the derivative of the sine function given by equation 5 is

$$(5) \qquad \frac{dy}{dx} = k\cos(k\,x)$$

The derivative is also estimated numerically using equations 3a, 3b and 3c. In calculating the derivatives using equations 3 at the end points, the forward approximation must be used for index 1 and the backward approximation must be used at index N.

The Matlab function **gradient** is also used to calculate the derivative with the code

```
dydxM = gradient(y,dx);
```

Figure 1 shows plots for the first derivative with the number of grid points being N = 101: Analytically exact result (A), Matlab gradient command (M), the forward (F), the backward (B) and central difference (C) approximations.

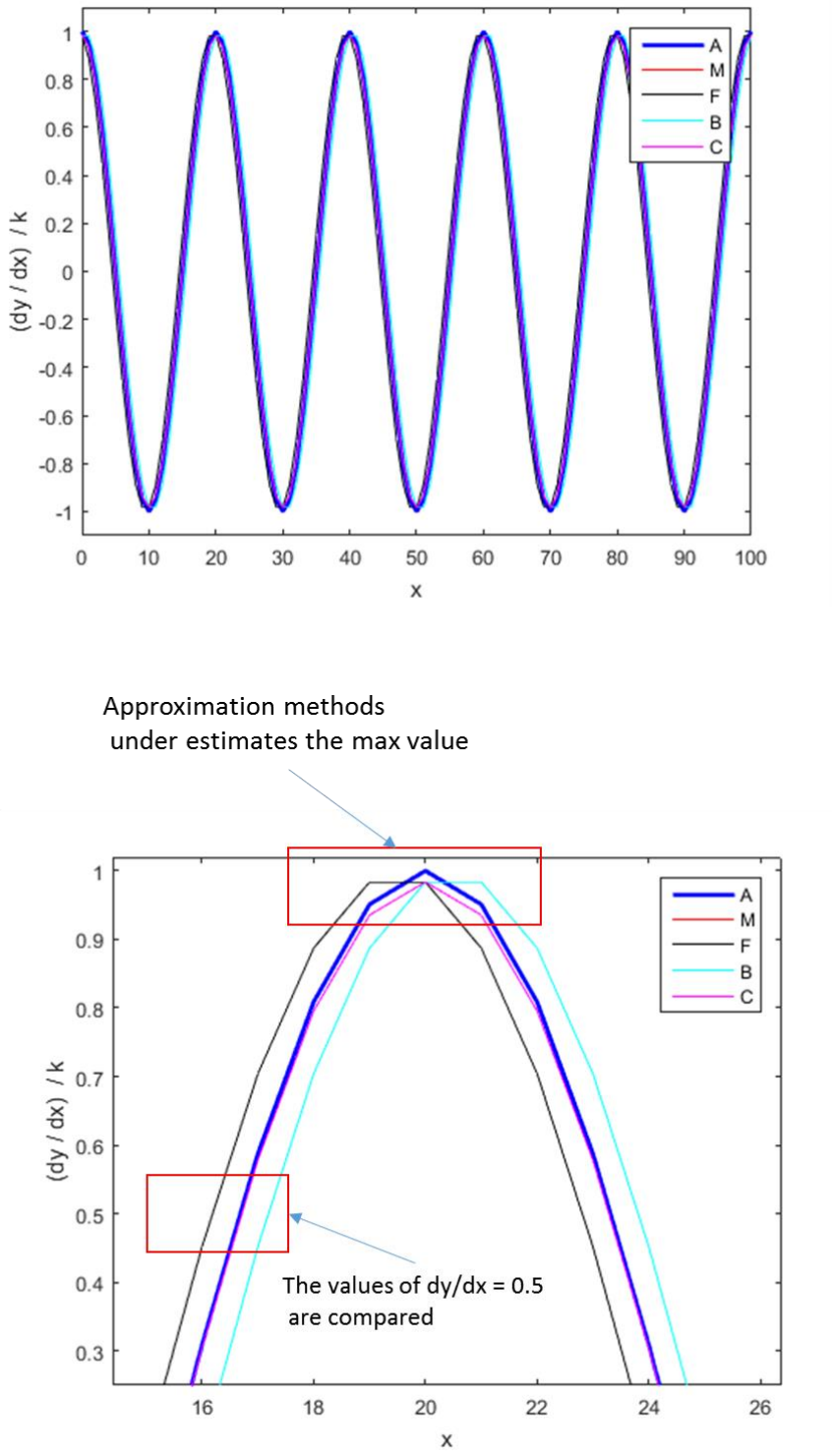


Fig 1. Plots of the first derivative for N = 101. **cemDiff01.m**

All the approximation methods under estimate the value of the derivative at the maximum values. To compare the accuracy of the different approximation methods we can't compare the values of the derivative at a maximum because all the methods predict the same result. So, to test the accuracy we will compare the ratio of the value of the dy/dx for each approximation method to the exact value of dy/dx at an x position when the exact value is equal to 0.5.

Table 1. Summary of the ratios for comparing the accuracy of the different methods.

N	Δx	Ratio A	Ratio M	Ratio F	Ratio B	Ratio C
101	1.0000	1.0000	0.9836	0.5042	1.4631	0.9836
201	0.5000	1.0000	0.9959	0.9389	1.0528	0.9958
501	0.2000	1.0000	0.9993	0.9913	1.0074	0.9993

From Table 1 it is clear that the central difference approximation method is better than the forward or backward method and it seems most likely that the Matlab gradient command uses a central difference method.

If you want to calculate the first derivative of a single variable function, then the simplest way is to use the Matlab gradient function

```
dydxM = gradient(y,dx)
```

Symbolic differentiation in Matlab

```
syms z t
% First Derivatives - symbolic functions to array values
 df_dz = diff(3*sin(2*z),z)   % w.r.t z
```

→ df_dz = 6*cos(2*z)

```
 u = 1:8;          % range
 v = subs(df_dz,u)   % symbolic  values
```

→ v = [6*cos(2), 6*cos(4), 6*cos(6), 6*cos(8), 6*cos(10), 6*cos(12), 6*cos(14), 6*cos(16)]

```
 v = double(v)      % numerical values
```

→ v = -2.4969 -3.9219 5.7610 -0.8730 -5.0344

5.0631 0.8204 -5.7460

```
 df_dz = diff(sin(3*z - 5*t),z)  % w.r.t  z
```

→ 3*cos(5*t - 3*z)

```
 df_dt = diff(sin(3*z - 5*t),t)  % w.r.t  t
```

→ -5*cos(5*t - 3*z)

```
 % Second Derivatives
  df2_dz2 = diff(sin(3*z - 5*t),z,2)  % w.r.t z
```

→ 9*sin(5*t - 3*z) = -9*sin(3*z - 5*t)

```
  df2_dt2 = diff(sin(3*z - 5*t),t,2)  % w.r.t t
```

→ 25*sin(5*t - 3*z) = -25*sin(3*z – 5*t)

SECOND DERIVATIVE OF A ONE VARIABLE FUNCTION

The second derivative of a function gives the rate of change of the first derivative or rate of change of the gradient of the function. It is easy to show that by using the central difference form for a derivative, the second derivative of $y(x)$ can be expressed as

$$\text{(6)} \qquad \frac{d^2y(x)}{dx^2} = \frac{y(x+\Delta x) - 2\,y(x) + y(x-\Delta x)}{2\,\Delta x}$$

Again, in writing the code to evaluate equation 6 you need to be careful at the end points (indices 1 and N). For index 1 use a forward difference approximation for the change in slope and for index N use a backward approximation.

Code for calculating the first derivative using equation 3c;

```
% Central difference approximation
     dydxC = zeros(1,N);
     dydxC(1) = (y(2)-y(1))/dx;
     dydxC(N) = (y(N)-y(N-1))/dx;
     for n = 2: N-1
         dydxC(n) = (y(n+1)-y(n-1))/(2*dx);
     end
```

Code for calculating the second derivative using equation 6;

```
    % Central difference approximation
     d2ydx2C = zeros(1,N);
      d2ydx2C(1) = (dydxC(2)-dydxC(1))/dx;
      d2ydx2C(N) = (dydxC(N)-dydxC(N-1))/dx;
      for n = 2: N-1
           d2ydx2C(n) = (y(n+1)-2*y(n) + y(n-1))/(dx^2);
      end
```

Code using the Matlab function gradient

```
     d2ydx2M = gradient(dydxM,dx);
```

However, the easiest way to find the second derivative of a single variable function is to use the **del2** function which corresponds to a Laplacian operator

```
     del2y = 4*del2(y,dx);
```

N.B. For the [1D] Laplacian function you need to multiply del2 by 4.

Figure 2 shows the plots of the second derivative of the function $y(x) = \sin(k\,x)$ for the exact solution (A), using the Matlab gradient function (Mgrad), the central difference approximation (C) and the del2 function (Mdel2).

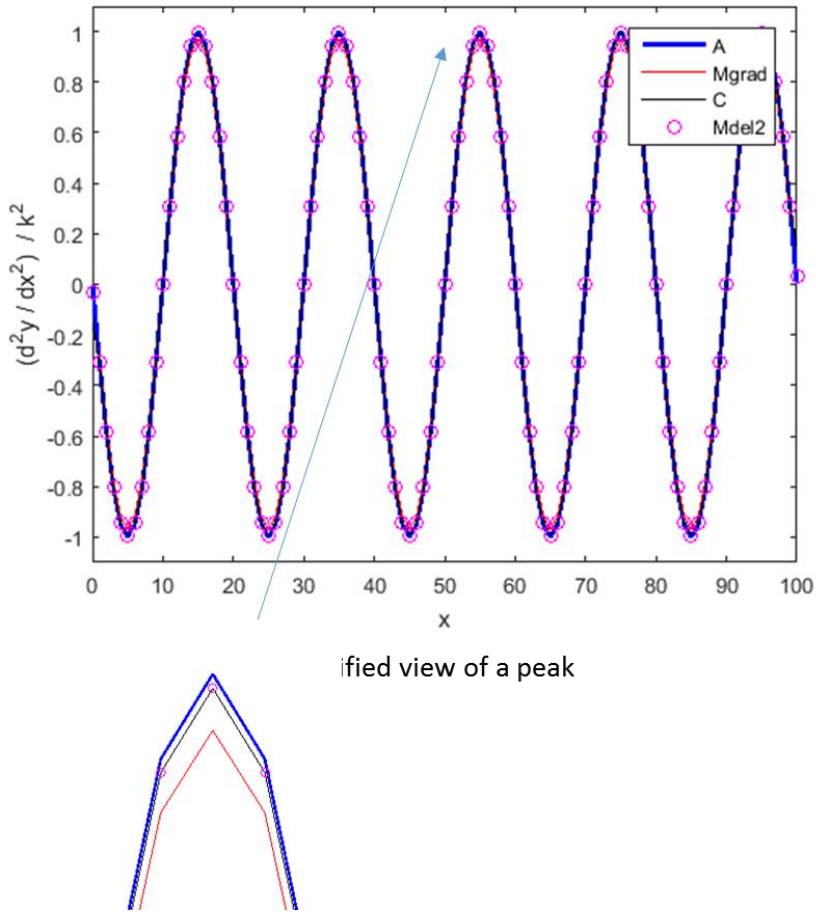


Fig.2 Second derivative $\dfrac{d^2y(x)}{dx^2}$ of the function $y(x) = \sin(k\,x)$ and a magnified view of one peak of the curve. The number of grid points is only N = 101. cemDiff01.m

From figure 2 the central difference approximation of equation 6 appears to be a more accurate estimation of the second derivative than applying the Matlab gradient function twice. The central difference method, equation 6 gives identical result as using the **del2** function.

The approximation for the first and second derivatives given by equations 3 and 6 are very important in the solution of differential using the finite difference method.

DIFFERENTIATION USING MATRICES

The process of differentiation can be considered as an operator acting upon a function. The operator **D** is represented by a square NxN matrix that acts on the function **y** given by a Nx1 column vector to give the derivative of the function **y'** also as an Nx1 column vector.

$$\begin{bmatrix} y'_1 \\ y'_2 \\ y'_3 \\ \dots \\ y'_{N-1} \\ y'_N \end{bmatrix} = \frac{1}{2\Delta x} \begin{bmatrix} -2 & 2 & 0 & \dots & 0 & 0 \\ -1 & 0 & 1 & \dots & 0 & 0 \\ 0 & -1 & 0 & \dots & 0 & 0 \\ \dots & & & & & \\ 0 & 0 & \dots & -1 & 0 & 1 \\ 0 & 0 & \dots & 0 & -2 & 2 \end{bmatrix} \begin{bmatrix} y_1 \\ y_2 \\ y_3 \\ \dots \\ y_{N-1} \\ y_N \end{bmatrix}$$

The matrix elements can be easily deduced from the definition of the first derivative

$$\frac{dy}{dx} \approx \frac{\Delta y}{\Delta x} = \frac{y(n+1) - y(n-1)}{2\Delta x} \quad D(1) = \frac{y(2) - y(1)}{\Delta x} \quad D(N) = \frac{y(N) - y(N-1)}{\Delta x}$$

Similarly, for the second derivative can also be represented as a matrix operation where the elements of the matrix are determined from the equation for the second derivative

$$\frac{d^2y}{dx^2} \approx \frac{y(n+1) - 2y(n) + y(n-1)}{\Delta x^2}$$

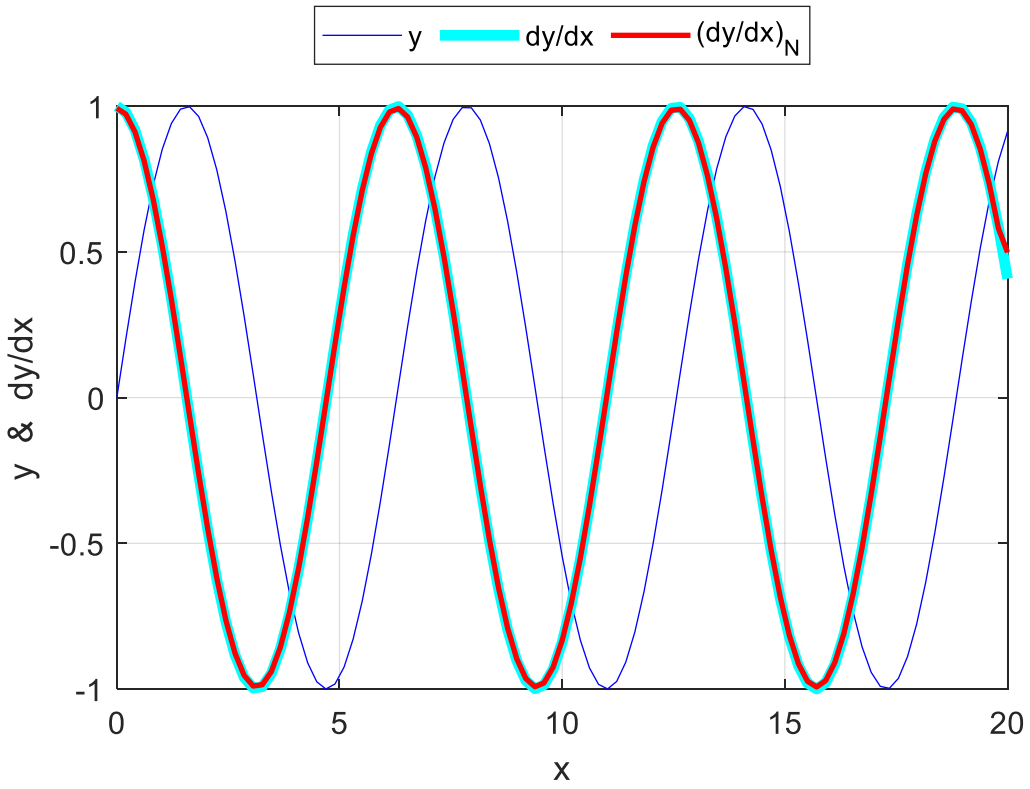
$$D^2(1) = \frac{y(3) - 2y(2) + y(1)}{\Delta x^2} \quad D^2(N) = \frac{y(N) - 2y(N-1) + y(N-1)}{\Delta x^2}$$

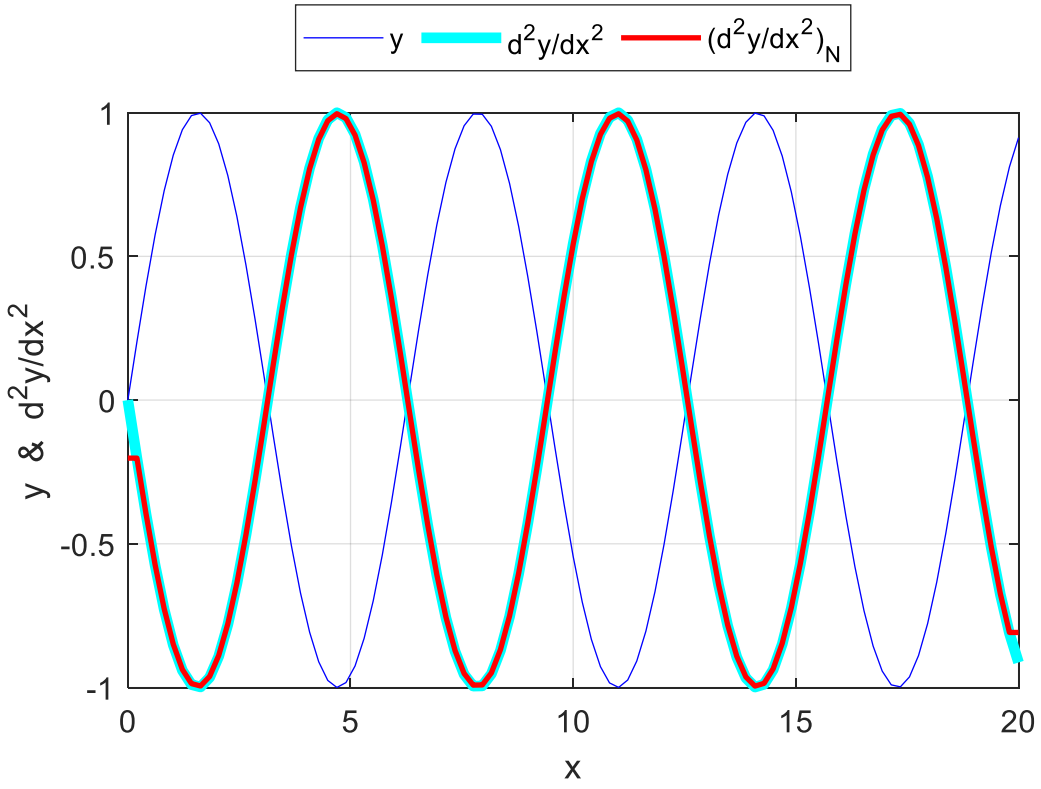
Matlab Script mqm002.m

```
% mqm002.m

% Calculation of the 1st and 2nd derivatives using a matrix for the
% first derivative operator and for the second derivative operator.

clear
close all
clc

% x grid
 N = 99;
 x = linspace(0,20,N)';
 dx = x(2) - x(1);

% Function y / first derivative v / second derivative a
 y = sin(x);
 v = cos(x);
 a = -sin(x);

% First dervative opertor matrix D and first derivative (numerical) dydx
 D = (diag(ones((N-1),1),1) - diag(ones((N-1),1),-1));
 D(1,1) = -2; D(1,2) = 2;
 D(N,N) = 2; D(N,N-1) = -2;
 dydx = D./(2*dx) * y;
```

```
% Second dervative opertor matrix D and second derivative (numerical)
% d2ydx2
  D = diag(ones((N-1),1),1) + diag(ones((N-1),1),-1) - 2.*diag(ones((N),1),0);
  D(1,1) = 1; D(1,2) = -2; D(1,3) = 1;
  D(N,N) = 1; D(N,N-1) = -2; D(N,N-2) = 1;
  d2ydx2 = D./dx^2 * y;



% GRAPHICS
============================================================
figure(1)
   set(gcf,'units','normalized');
   set(gcf,'Position', [0.05 0.05 0.30 0.35])
   set(gcf,'color','w');
   FS = 12;


    xP = x; yP = y;
    plot(xP,yP,'b');
    hold on
    yP = v;
    plot(xP,yP,'c','linewidth',4)
    yP = dydx;
    plot(xP,yP,'r','linewidth',2)
    grid on
    xlabel('x')
```

```
    ylabel('y  &  dy/dx')
    set(gca,'fontsize',FS)
    legend('y','dy/dx', '(dy/dx)_N','location','northoutside',...
       'Orientation','horizontal')

figure(2)
   set(gcf,'units','normalized');
   set(gcf,'Position', [0.05 0.45 0.30 0.35])
   set(gcf,'color','w');
   FS = 12;

    xP = x; yP = y;
    plot(xP,yP,'b');
    hold on
    yP = a;
    plot(xP,yP,'c','linewidth',4)
    yP = d2ydx2;
    plot(xP,yP,'r','linewidth',2)
    grid on
    xlabel('x')
    ylabel('y  &  d^2y/dx^2')
    set(gca,'fontsize',FS)
    legend('y','d^2y/dx^2', '(d^2y/dx^2)_N','location','northoutside',...
       'Orientation','horizontal')
```


y
dy/dx
(dy/dx)N
y & dy/dx
x
1
0.5
0
-0.5
-1
0
5
10
15
20


y
d2y/dx2
(d2y/dx2)N
y & d2y/dx2
x
1
0.5
0
-0.5
-1
0
5
10
15
20

GRADIENT OF A SCALAR FIELD / DEL OPERATOR ∇

Consider a **scalar field** defined as by scalar function of four variables of space (x, y, z) and time (t) such as temperature $T(x, y, z, t)$. We can differentiate with respect one of the variables, keeping the other three constant. The result is the partial derivative. The partial derivative w.r.t x is

$$\frac{\partial T}{\partial x} = \lim_{x \to 0} \left(\frac{T(x + \Delta x, y, z, t) - T(x, y, z, t)}{\Delta x} \right)$$

If we permit all variables to vary, we get the total derivative form by the chain rule, for example the total derivative for time is

$$(7) \qquad \frac{dT}{dt} = \frac{\partial T}{\partial x}\frac{dx}{dt} + \frac{\partial T}{\partial y}\frac{dy}{dt} + \frac{\partial T}{\partial z}\frac{dz}{dt}$$

Like vector quantities, the differentiation of a vector can be carried out with respect to one or more of the variables of the vector, for example,

$$(8) \qquad \frac{\partial \vec{V}}{\partial t} = \frac{\partial}{\partial t}\left(\hat{i} V_x + \hat{j} V_y + \hat{k} V_z \right) = \hat{i}\frac{\partial V_x}{\partial t} + \hat{j}\frac{\partial V_y}{\partial t} + \hat{k}\frac{\partial V_z}{\partial t}$$

The above derivatives have their uses, but, of greater interest is the operator called **del**

$$(9a) \qquad \nabla \equiv \hat{i}\frac{\partial}{\partial x} + \hat{j}\frac{\partial}{\partial z} + \hat{k}\frac{\partial}{\partial z} \qquad \text{operator called } \textbf{del}$$

Equation 9 is the del operator in Cartesian coordinates. The gradient in cylindrical coordinates (ρ, ϕ, z) and spherical coordinates (r, ϕ, θ) is given by

$$(9b) \qquad \nabla f = \hat{\rho}\frac{\partial f}{\partial \rho x} + \hat{\phi}\frac{1}{\rho}\frac{\partial f}{\partial \phi} + \hat{k}\frac{\partial f}{\partial z} \qquad \text{cylindrical coordinates}$$

$$(9c) \qquad \nabla f = \hat{r}\frac{\partial f}{\partial r} + \hat{\phi}\frac{1}{r\sin\theta}\frac{\partial f}{\partial \phi} + \hat{\theta}\frac{1}{r}\frac{\partial f}{\partial \theta} \qquad \text{spherical coordinates}$$

The gradient of a scalar function f is the vector function

$$(10) \qquad \nabla f = \hat{i}\frac{\partial f}{\partial x} + \hat{j}\frac{\partial f}{\partial z} + \hat{k}\frac{\partial f}{\partial z} \qquad \textbf{gradient}$$

The **gradient** of the scalar function f is the vector whose magnitude at any point is the maximum space rate of change of the function and the direction of the vector points in the direction of the **maximum increase in the function** f.

The mscript **cemDiff02.m** can be used to give [3D] plots for a scalar function of the form $f(x,y)$ and to calculate and display the gradient ∇f. As an example, we will take the function $f(x,y) = xe^{-x^2-y^2}$. Figure 3 shows plots of the scalar field for $f(x,y)$ and the vector field for ∇f as a set of arrows. At the position of each arrow, the magnitude of the vector field is proportional to the length of the arrow and direction of the field by the direction of the arrow. The direction of the arrow points in the direction of the maximum increase in slope of the scalar field at that point. Comparing the upper and lower plots, you can see that the arrows always points "uphill". The arrows are also perpendicular to the contour lines.

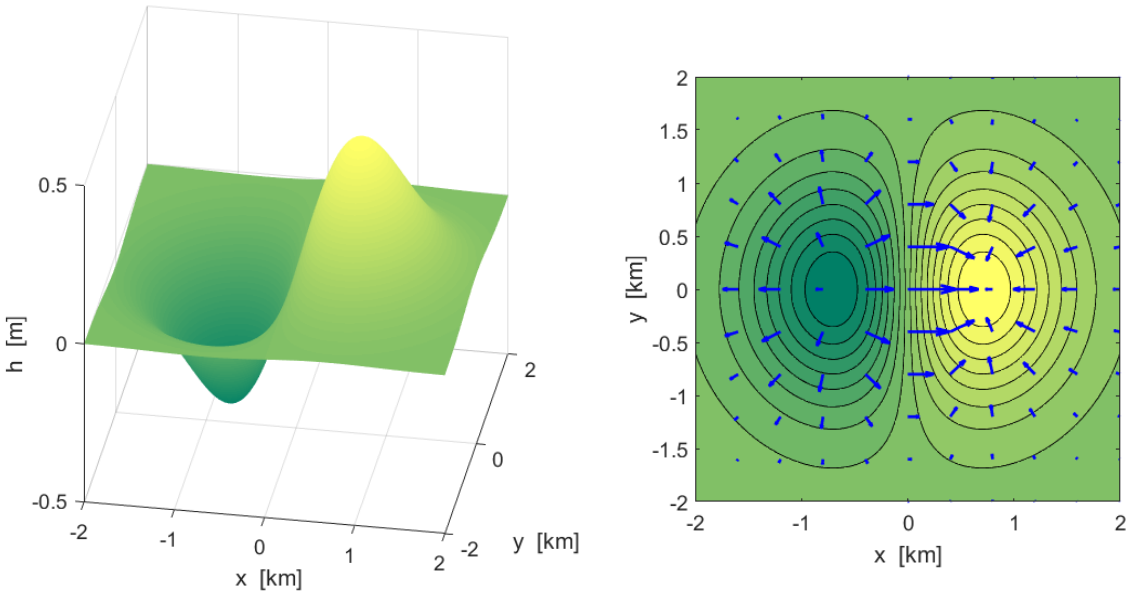


Fig. 3. [3D] plots of the function $f(x,y) = xe^{-x^2-y^2}$ using the surf function and the contourf function. The vector field ∇f is given by the arrows using the quiver function. cemDiff02.m

Example D.J Griffiths *Introduction to Electrodynamics* (Problem 1.12)

The height of a certain hill [m] is given by

$$h = 10\left(2xy - 3x^2 - 4y^2 - 18x + 28y + 12\right)$$

where +x [km] is the distance to the east and +y [km] is the distance to the north of a reference point and h is the height with respect to the reference point.

(A) What is the location of the top of the hill?

(B) How high is the hill with respect to the reference point?

(C) How steep is the slope of the hill [m/km] at the position 30 km south and 20 km east of the reference location? In what direction is the slope steepest at this position? How high is the hill at this location?

We will use Matlab to find the answers to this question.

For the Matlab mscript we will use unrealistic numbers of 100 km x 100 km for the region surrounding the hill and reference point to better understand the properties of the gradient operator.

$\Rightarrow$

Using Matlab we can explore the answers to this problem in much more detail that the traditional pen and paper method. We can write a Matlab mscript to compute the height function h and its gradient ∇h. We can plot the results to better visualise the problem and to gain a better understanding of the gradient operator. To answer the questions, we need to use Matlab logical functions to get the numerical results.

Matlab tutorial ⇒

You have to be careful interpreting logical functions when using the [2D] arrays used for plotting. Before we answer the question above, we will consider a short tutorial on arrays and logical functions.

Consider the function $z(x,y) = x + y$. The x and y values are given by the 5 element arrays

$x = [10\ 20\ 30\ 40\ 50]$ $\quad\quad$ $y = [11\ 12\ 13\ 15\ 20]$

We want to find the value of z when x = 20 and y = 15 from the [2D] array for z.

Matlab Command Window – finding the index of an array

```
x = [10 20 30 40 50]          y = [11 12 13 15 20]
nx = min(find(x>=20))   →   nx = 2
ny = min(find(y>=14))   →   ny = 4
[xx yy] = meshgrid(x,y)
zz = xx + yy   →
```

xx =					yy =					zz =				
10	20	30	40	50	11	11	11	11	11	21	31	41	51	61
10	20	30	40	50	12	12	12	12	12	22	32	42	52	62
10	20	30	40	50	13	13	13	13	13	23	33	43	53	63
10	20	30	40	50	15	15	15	15	15	25	**35**	45	55	65
10	20	30	40	50	20	20	20	20	20	30	40	50	60	70

The x values are the columns of xx and the rows of yy give the y values.

We can now find the value of zz when x = 20 and y = 15. The required element of the array zz has the **column index is nx** = **2** and the **row index is ny** = **4**

zz(**4**,**2**) → 35 $\quad\quad$ (x+y = 20+15 = 35)

N.B. the x index gives column number and the y index gives the row number

⇐

The Matlab mscript **cemDiff02.m** is used for the plots and to find the numerical answers for the HILL problem.

For the plots we need to form a [2D] grid using the meshgrid function

```
[xx, yy] = meshgrid(x,y);
```

where x and y are the grid point along the X axis and Y axis respectively. The x and y variables both go from - 50 km to +50 km.

xx is the [2D] array for the x positions and yy is the [2D] array for the y positions of the grid.

xx → columns yy → rows

The function for the height is

```
f = 10 .* (2.*xx.*yy - 3.*xx.^2 -4.*yy.^2 - 18.*xx + 28.*yy +12);
```

and the gradient of the function is

```
[delx, dely] = gradient(f,dx,dy);
```

where $delx$ and $dely$ are the x and y components of the vector for the gradient of the function f.

Figure 4a is a [3D] plot of the hill $h(x, y)$ and figure 4b shows a [2D] contour plot of the hill and the arrows show the gradient ∇h. The vector ∇h points in the direction of the arrow and has a magnitude that is proportional to the length of the arrow. At each location of an arrow, the arrow points in the direction of the maximum increase in the value of h. This is why all the arrows point up the hill towards its apex and are directed at right angles to the contour lines.

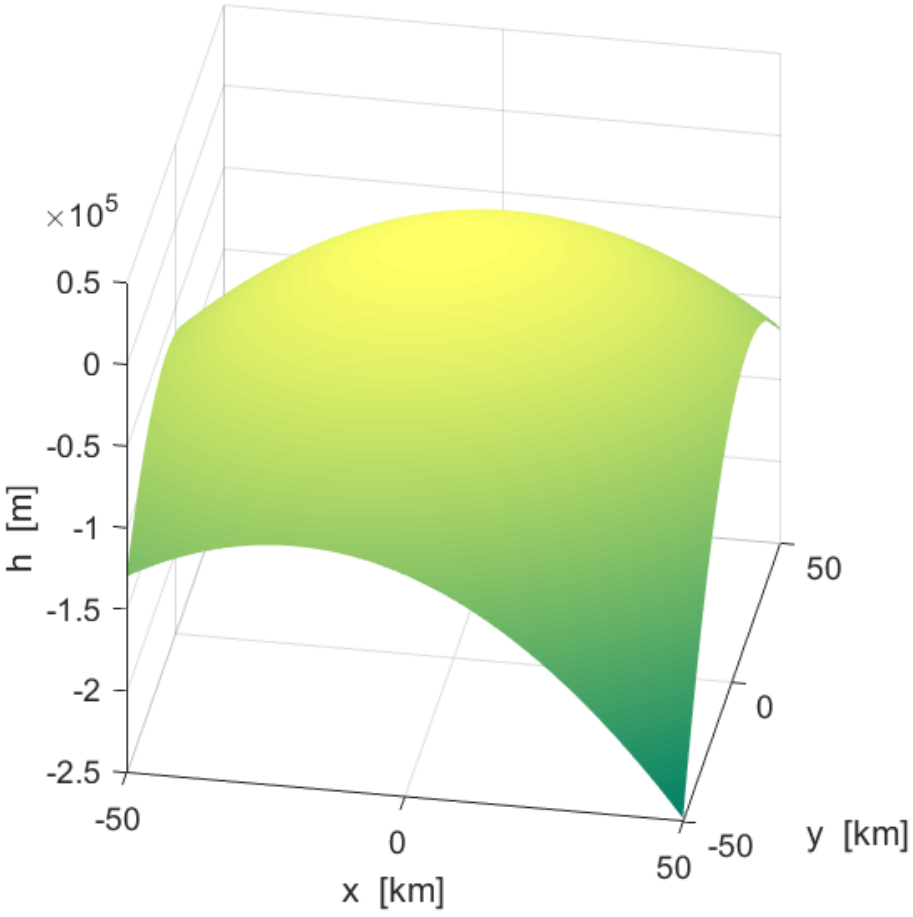


Fig. 4a. [3D] plot of the hill $h(x, y)$. The plot was generated using the surf function `surf(xx,yy,f); shading interp;`

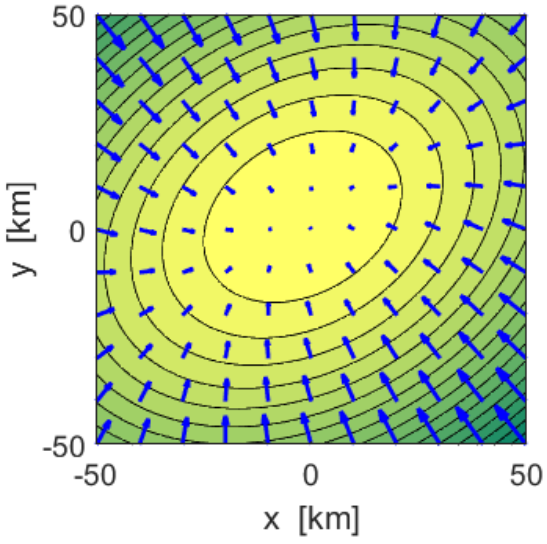


Fig. 4b. Contour plot of the hill using the contourf function
`contourf(xx, yy, f, 16);`

The arrows representing the gradient are plotted using the quiver function

```
d = 1:10:N;
p1 = xx(d,d); p2 = yy(d,d); p3 = delx(d,d);
p4 = dely(d,d);
h = quiver(p1, p2, p3, p4);          cemDiff01.m
```

(A) The height of the hill occurs when $\nabla h = 0$ or when h is a maximum.

The (x, y) coordinates of the top of the hill can be found from the Matlab Command Window

`[a b] = find(f == max(max(f)))` → a = 54 b = 49

`xx(54,49)` → x = -2

`yy(54,49)` → y = 3

where a is the row index and b the column index for the [2D] arrays. So, the top of the hill is located -2 km to the west and 3 km to the north of the reference location.

(B) The height of the hill is found from its (x, y) location

`f(54,49)` → f = 720

The height of the hill is 720 m.

(C) We need to find the gradient and the height at the location where $x = +20$ and $y = -30$. To do this we need to find the array elements for the height and gradient. Let $\nabla h = \vec{G} = G_x \hat{i} + G_y \hat{k}$

Matlab Command Window

`nx = min(find(x>=20))` → nx = 71

`ny = min(find(y>=-30))` → ny = 21

`x(nx)` → x = 20

`y(nx)` → y = -30

`xx(ny,nx)` → xx = 20

`xy(ny,nx)` → yy = -30

`f(ny,nx)` → h = -71880

`delx(ny,nx)` → Gx = -1980

`dely(ny,nx)` → Gy = 3080

`G = sqrt(Gx^2+Gy^2)` → G = 36615

`A = atand(abs(Gy/Gx))` → A = 57.3

The location 30 km south and 20 km east of the reference location:

Height of hill = 71880 m lower than the reference point.

The slope of the hill = 36615 m/km

The direction of steepest ascent = 57.3° N of W

DIVERGENCE $\nabla \bullet \vec{V}$

The quantity $\nabla \bullet \vec{V}$ is known as the **divergence**. The divergence of a vector field is a scalar field (divergence of a scalar quantity has no meaning). The vector field is specified by a vector $\vec{V}$ and the scalar field specified by its divergence $\nabla \bullet \vec{V}$.

Equation (9a) gives the equation for the operator del Cartesian components.

(9a) $$\nabla \equiv \hat{i}\frac{\partial}{\partial x} + \hat{j}\frac{\partial}{\partial z} + \hat{k}\frac{\partial}{\partial z}$$ operator called **del**

The divergence is the dot product of the del operator and a vector $\vec{V}$

$$\vec{V} = \hat{i}V_x + \hat{j}V_y + \hat{k}V_z$$

(11a) $$\nabla \bullet \vec{V} = \frac{\partial V_x}{\partial x} + \frac{\partial V_y}{\partial y} + \frac{\partial V_z}{\partial z}$$ Cartesian coordinates

(11b) $$\nabla \bullet \vec{V} = \frac{1}{\rho}\frac{\partial\left(\rho_x V_\rho\right)}{\partial \rho} + \frac{1}{\rho}\frac{\partial V_\phi}{\partial \phi} + \frac{\partial V_z}{\partial z}$$ cylindrical coordinates

(11c) $$\nabla \bullet \vec{V} = \frac{1}{r^2}\frac{\partial\left(r^2 V_r\right)}{\partial r} + \frac{1}{r\sin\theta}\frac{\partial\left(\sin\theta V_\theta\right)}{\partial \theta} + \frac{1}{r\sin\theta}\frac{\partial V_\phi}{\partial \phi}$$

spherical coordinates

The divergence tells us how the vector $\vec{V}$ field spreads out (diverges) at the point in question. You can think of a positive divergence as a source (tap) and a negative divergence as a sink (drain). Consider the two-dimensional example of water flowing. The vector field given by $\vec{V}$ gives the magnitude and direction of the water flow at each point in the field. For example, at a point on the water surface you scatter a handful of saw dust, if the saw dust spreads out then the divergence is positive, if the saw dust collects together, then the divergence is negative.

When $\nabla \bullet \vec{V} = 0$ everywhere, the vector field $\vec{V}$ is called **solenoidal**.

THE CURL OF A VECTOR FUNCTION $\nabla \times \vec{V}$

The curl of a vector field is written as $\nabla \times \vec{V}$. The curl is a vector that results from the del operator acting upon a vector like the cross product of two vectors

$$\vec{V} = \hat{i}V_x + \hat{j}V_y + \hat{k}V_z$$

(12a) $$\nabla \times \vec{V} = \begin{vmatrix} \hat{i} & \hat{j} & \hat{k} \\ \partial/\partial x & \partial/\partial x & \partial/\partial x \\ V_x & V_y & V_z \end{vmatrix}$$ Cartesian coordinates

Where the curl is nonzero, the vector field of the curl has a sort of rotational symmetry. A nonzero curl results from vector field points in one direction and increases in another direction. If the vector field points in a given direction and increases in that the direction then the curl is zero. Hence, the curl is related to how the vector field changes as you move across the field.

$\nabla \times \vec{V}$ is a measure of how much the vector field $\vec{V}$ "curls around" the point in question.

The divergence provides information about the change in the vector field as you move along the field, whereas, the curl does the same across the field.

When $\nabla \times \vec{V} = 0$ everywhere the vector field is called **irrotational**.

It is meaningless to talk about the curl of a scalar quantity.

Matlab **cemDiff03.m** and **cemDiff04.m**

```
divV = divergence(xx,yy,zz,Vx,Vy,Vz)
curlV = curl(xx,yy,zz,Vx,Vy,Vz)
```

Download the files **cemDiff03.m** and **cemDiff04.m** and examine the code so that you understand each step in calculating the divergence and curl and how to visualising the vector, divergence and curl fields. The div and curl are calculated at all the points given by the arrays xx, yy and zz for the vector $\vec{V}$ which has components $\left(V_x, V_y, V_z\right)$.

The range for the [3D] fields is specified by the limits for the x, y and z values

```
%INPUTS
=============================================================
% Number of grid points (integer)
  N = 101;
% Range for X and Y values  [minX maxX minY maxY minZ maxZ]
  XY = [-1 1 -1 1 -1 1];
```

The [3D] grid is formed using the **meshgrid** function

```
 minX  = XY(1);  maxX = XY(2);
 minY  = XY(3);  maxY = XY(4);
 minZ  = XY(5);  maxZ = XY(6);
 x = linspace(minX, maxX,N);
 y = linspace(minY, maxY,N);
 z = linspace(minZ, maxZ,N);
 [xx, yy, zz] = meshgrid(x,y,z);
```

For the vector field $\vec{V} = \hat{i}V_x + \hat{j}V_y + \hat{k}V_z = \hat{i}\,xy + \hat{j}\,y^2 + \hat{k}\,0$

```
Vxx = xx .* yy;  Vyy = yy.^2;  Vzz = 0 .* Vxx;
```

The divergence and curl of the vector field $\vec{V}$

```
divV = divergence(xx, yy, zz, Vxx, Vyy, Vzz);
```

```
[curlVxx, curlVyy, curlVzz] = curl(xx, yy, zz, Vxx, Vyy, Vzz);
```

The x, y, and z values for the field must have the same number of elements and must be monotonic, but do not need to be uniformly spaced.

The mscript **cemDiff03.m** is used to calculate the divergence for [2D] vector fields. In running this Script you only need to enter the number of grid points, the range for the x and y values and define the vector function. The default vector field is defined by the sinusoidal functions

$$V_x = \cos\left(\frac{2\pi x}{\lambda_x}\right) \qquad V_y = \cos\left(\frac{2\pi y}{\lambda_y}\right)$$

where $\lambda_x = 50$ and $\lambda_y = 40$. The range for the x and y values is from -50 to +50.

Figure 5 shows the vector field as a set of blue arrows. The length of an arrow at a point is proportional to the strength of the vector field at that point and the arrow points in the direction of the vector field. The background shading shows the scalar field for the divergence. A [3D] plot of the scalar field for the divergence is shown in figure 6.

The arrows are produced using the **quiver** function

```
p1 = xx(d,d); p2 = yy(d,d); p3 = Vxx(d,d); p4 = Vyy(d,d);
h = quiver(p1, p2, p3, p4);
set(h,'color',[0 0 1],'linewidth', 2);
```

and the shading using the **pcolor** function

```
pcolor(xx,yy,div);
shading interp
colorbar
```

The [3D] plot of figure 6 was produced using the **surf** function

```
surf(xx,yy,div);
shading interp;
colormap(summer)
```

By examination of figures 5 and 6 it clearly demonstrates the physical significance of the divergence of a vector field: when the arrows are converging than the divergence is a negative and when the arrows diverge, the divergence is positive.

You can examine numerical values in figures 5 and 6 using the data cursor tool. Numerical results can be found in the Command Window, for example,

x(20) → -31 y(20) → -31

div(20,20) → -0.2403 Vxx(20,20) = -0.7290 Vyy(20,20) → 0.1564

You could also modify the program to include logical function to compute the vector field and divergence at any point in the fields.

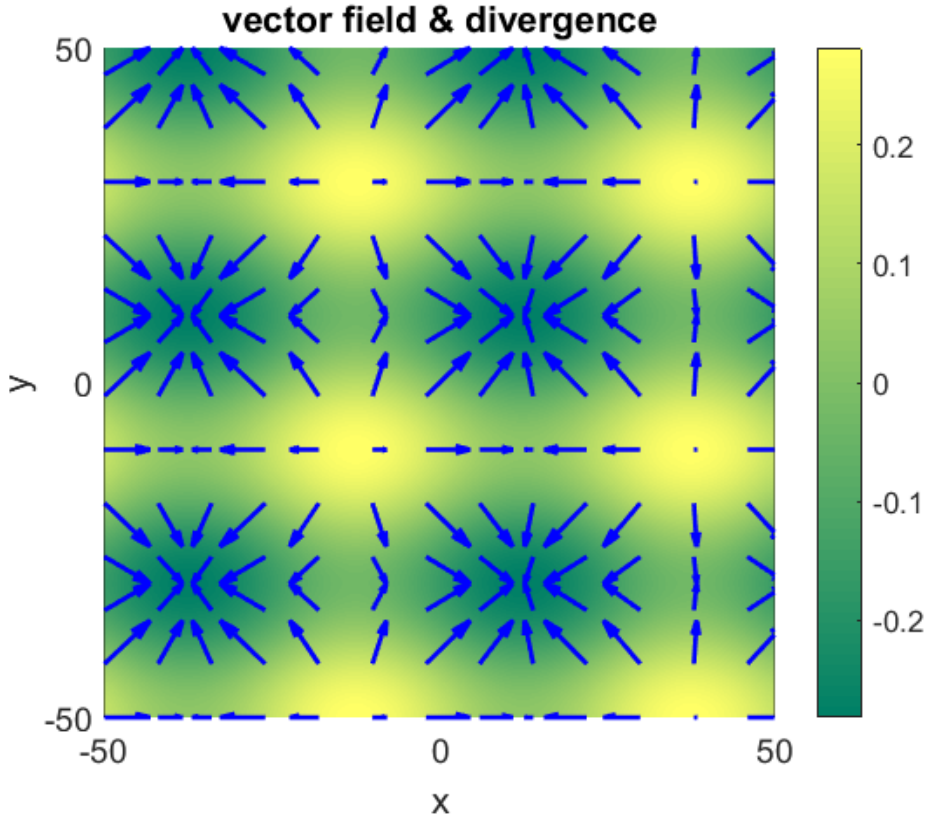


Fig. 5. The blue arrows show the vector field and the background shading the scalar field for the divergence. A converging vector field gives a negative divergence and a diverging vector field has a positive divergence. **cemDiff03.m**

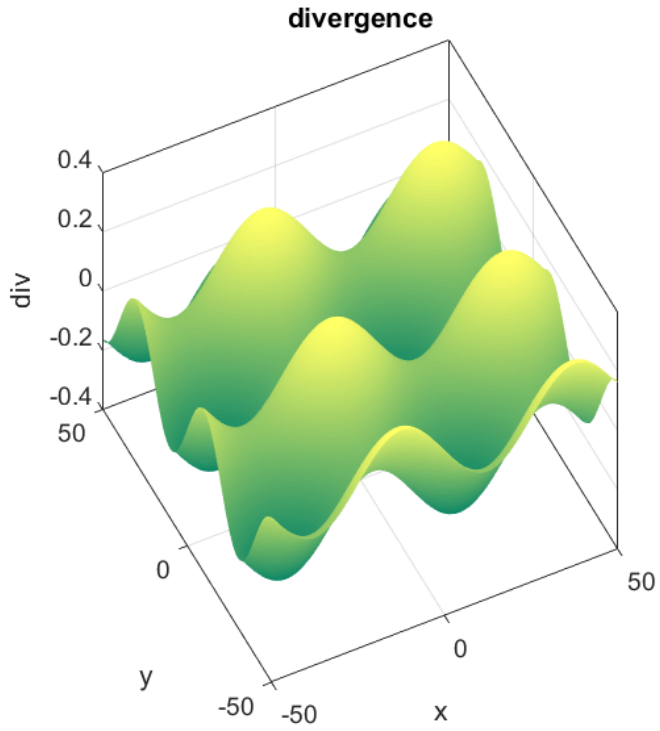


Fig. 6. A [3D] plot of the scalar field for the divergence. **cemDiff03.m**

It is easy to modify the mscript **cemDiff03.m** to investigate the divergence of other [2D] vector fields.

The mscript **cemDiff04.m** is used to calculate the divergence and curl for [3D] vector fields. In running this Script you only need to enter the number of grid points, the range for the x, y and z values and define the vector function.

It is more difficult to visualise the fields in [3D] compared to [2D] fields. Figure 7 shows the [3D] plot for the vector field $\vec{V} = \hat{i}\,x\,y + \hat{j}\,y^2 + \hat{k}\,0$ using the **quiver3** function

```
dx = 1:10:N; dy  = dx;   dz = dx;
p1 = xx(dx,dy,dz); p2 = yy(dx,dy,dz); p3 = zz(dx,dy,dz);
p4 = Vxx(dx,dy,dz); p5 = Vyy(dx,dy,dz);
p6 = Vzz(dx,dy,dz);
h = quiver3(p1, p2, p3, p4, p5, p6);
set(h,'color',[0 0 1],'linewidth', 2);
```

The index variables dx, dy and dz specify the array elements that are plotted as arrows for the **quiver3** plot. Eleven XY planes are shown in the Z direction. To view only single XY plane set the value dz to an integer corresponding to the index of the required z value as shown in figure 8.

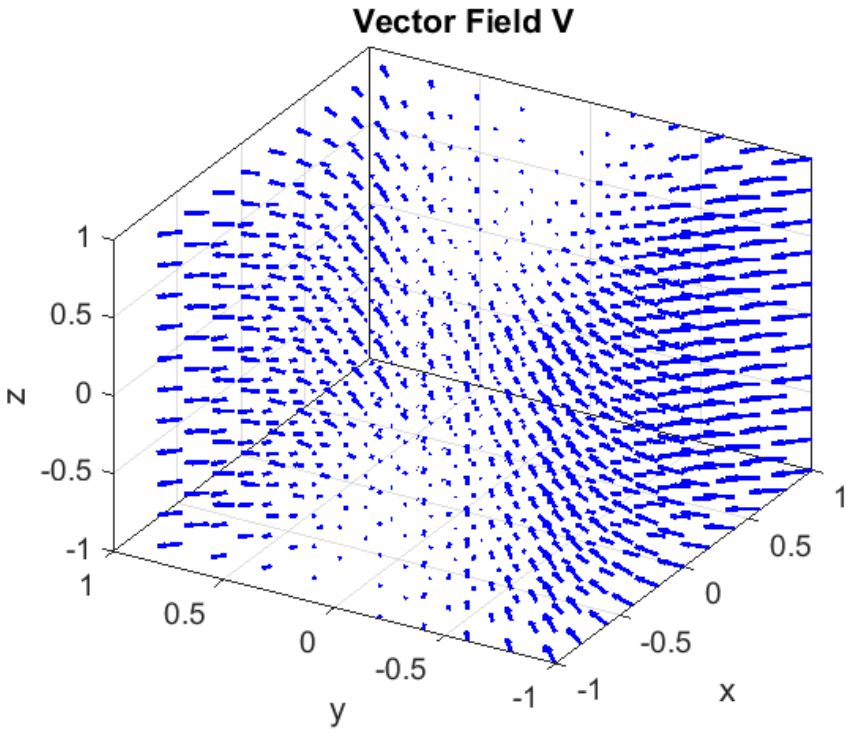


Fig.7. [3D] plot of the vector field in eleven XY planes using the **quiver3** function. **cemDiff04.m**

Figure 8 displays the vector function in the XY plane z = -1 (z-index dz = 1), the divergence and the curl of the vector field

$$\vec{V} = \hat{i}\, x\, y + \hat{j}\, y^2 + \hat{k}\, 0$$

divergence $\nabla \bullet \vec{V} = 3y$

curl $\nabla \times \vec{V} = -x\, \hat{k}$

The divergence is independent of the x and z values. The value of the divergence is proportional to the value of y. When y < 0 then the vector field converges and when y > 0, the vector field divergences. The curl is independent of the y and z values. The curl is directed in the positive Z direction when x > 0 and directed in the – Z direction for x < 0 (right hand screw rule in an XY plane: x > 0 arrows (fingers) curl anticlockwise and thumb points in +Z direction and for x < 0, arrows (fingers) curl clockwise and the thumb points in the – Z direction. Upon careful examination of the plots in figure 8, you can verify the behaviour of the vector, divergence and curl fields.

Numerical values can be obtained using the Data Cursor in the Figure Windows or by typing commands in the Command Window.

You need to be careful about in the interpretation and indexing (n_1,n_2,n_3) of the [3D] arrays in Matlab, for example,

xx(9,**40**,70) → -0.2200 xx(10,40,70) → -0.2200 xx(9,40,71) → -0.2200

xx(9, 41, 70) → -0.2000 x(**40**) → -0.2200

the index **n_2** changes the value of x and not the index n_1

yy(**9**,40,70) → -0.8400 yy(10,40,70) → -0.8200 y(**9**) = -0.8400

the index **n_1** changes the value of y and not the index n_2

zz(9,40,**70**) → 0.3800 zz(10,40,71) → 0.4000 z(**70**) = 0.3800

the index n_3 changes the value of z

Vxx(9,40,70) → 0.1848 $V_x = x\,y = (-0.2200)(-0.8400) = 0.1848$ √

Vyy(9,40,70) → 0.7056 $V_y = y^2 = (-0.8400)^2 = 0.7056$ √

Vzz(9,40,70) → 0 $V_y = 0$ √

divV(9,40,70) → -2.5200 $\nabla \bullet \vec{V} = 3\,y = (3)(-0.8400) = -2.5200$ √

curlxx(9,40,70 → 0 √

curlyy(9,40,70) → 0 √

curlzz(9,40,70) → 0.2200 $\nabla \times \vec{V} = -x\,\hat{k} = +0.2200\,\hat{k}$ √

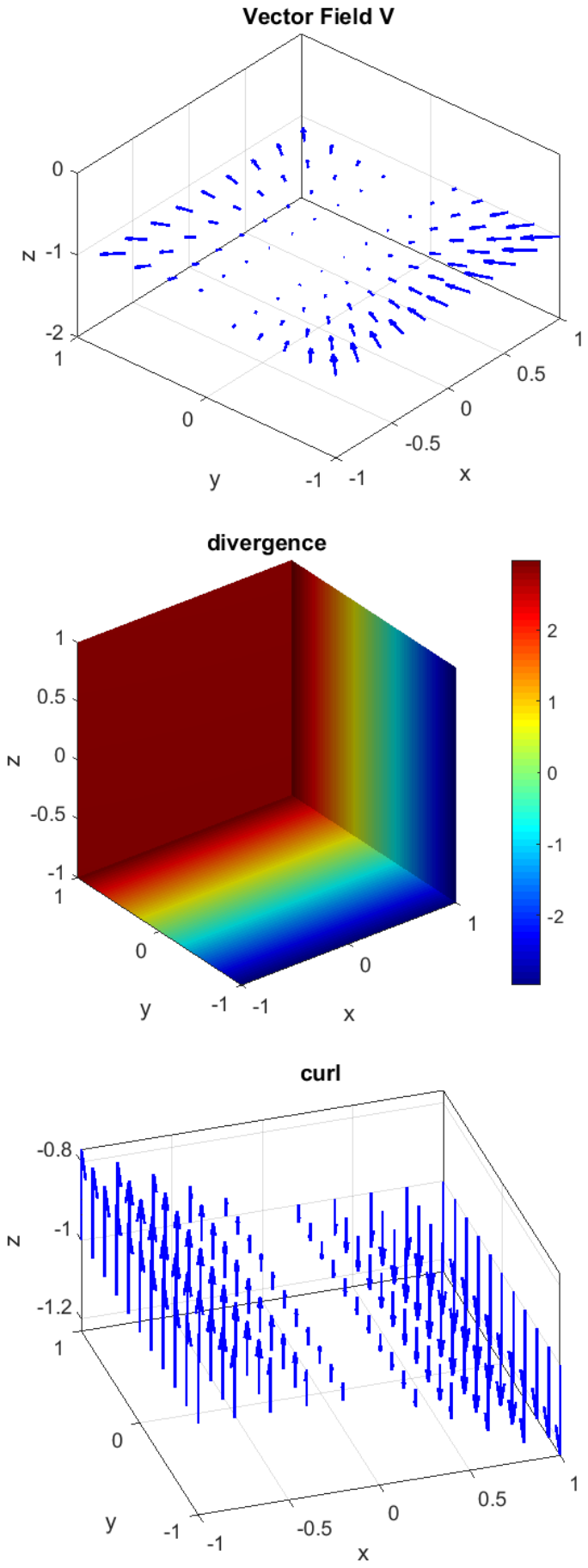


Fig. 8. Divergence and curl of the vector field $\vec{V} = \hat{i}\, x\, y + \hat{j}\, y^2 + \hat{k}\, 0$.

It is a simple matter to change the function for the vector field and to explore its divergence and curl. For example, $\vec{V} = \hat{i}(-R\sin\theta) + \hat{j}(R\cos\theta) + \hat{k}0$

```
rr = sqrt(xx.^2 + yy.^2);
tt = atan2(yy,xx);
Vxx = -rr.*sin(tt);
Vyy = rr.*cos(tt);
Vzz = 0.*zz.^2;
```

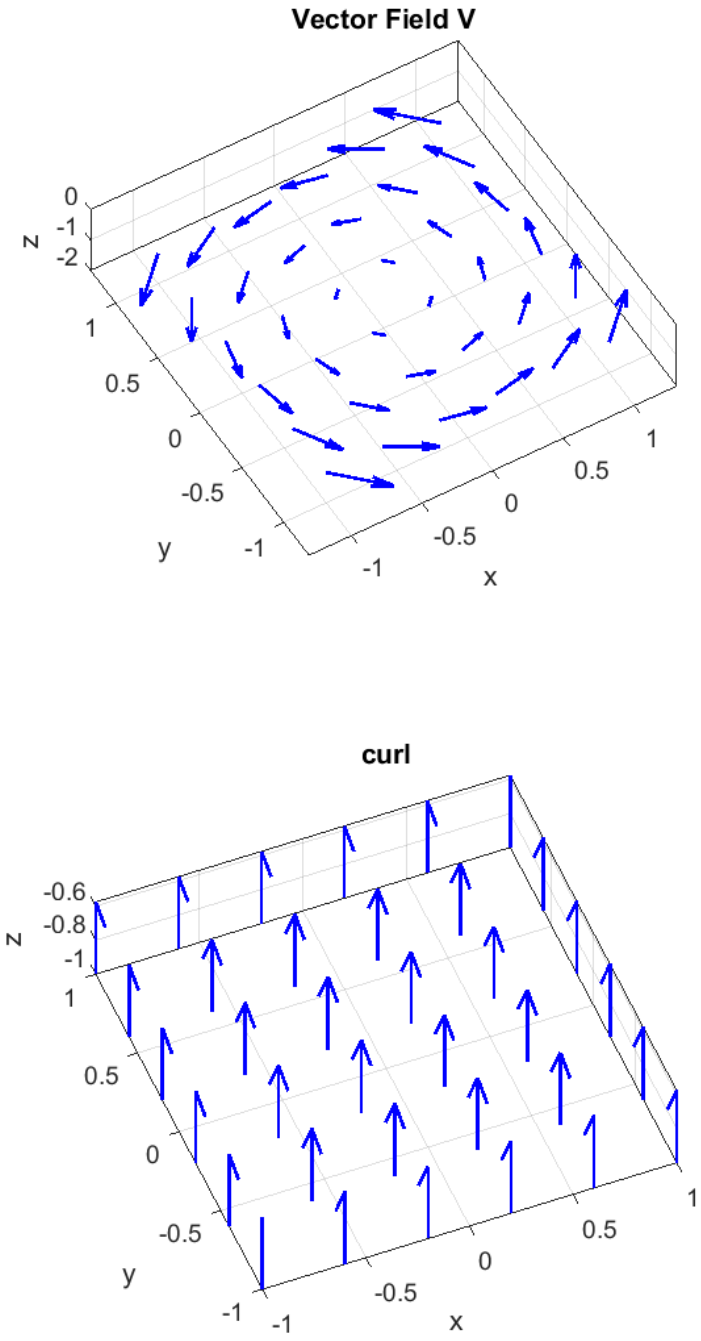


Fig.9. A vector field and its curl. Right hand screw rule gives the direction of the curl. **cemDiff04.m**

We will now consider the **slice** function in more detail for viewing the divergence of the vector field $\vec{V} = \hat{i}\left(y^2\right) + \hat{j}\left(2xy + z^2\right) + \hat{k}\left(2yz\right)$

slice(xx,xx,zz,divV,[x_ slices], [y_ slices], [z_ slices])

The **slice** function gives a [3D] coloured plot of the function div in the perpendicular planes specified by the arrays **[x_ slices], [y_ slices], [z_ slices]**.

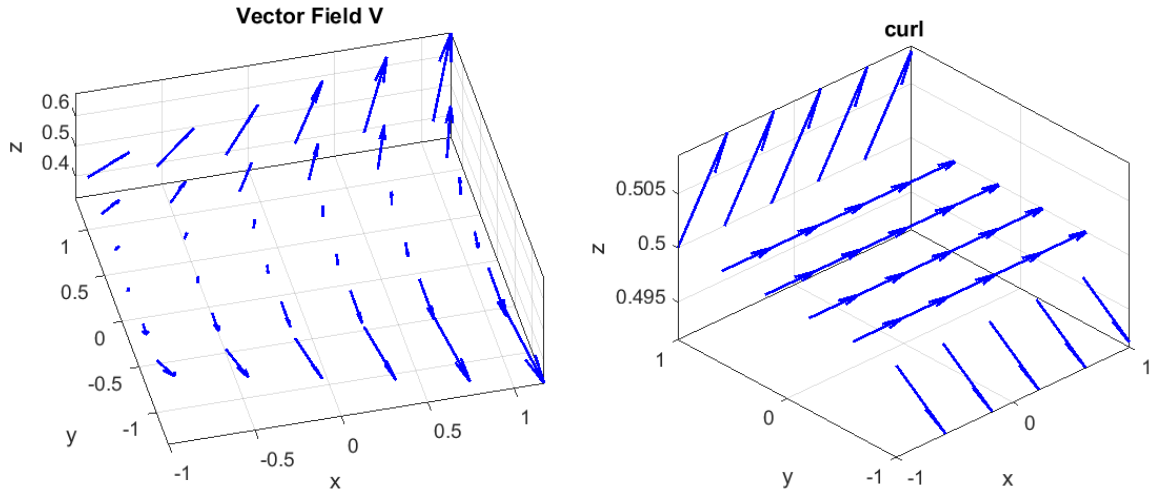
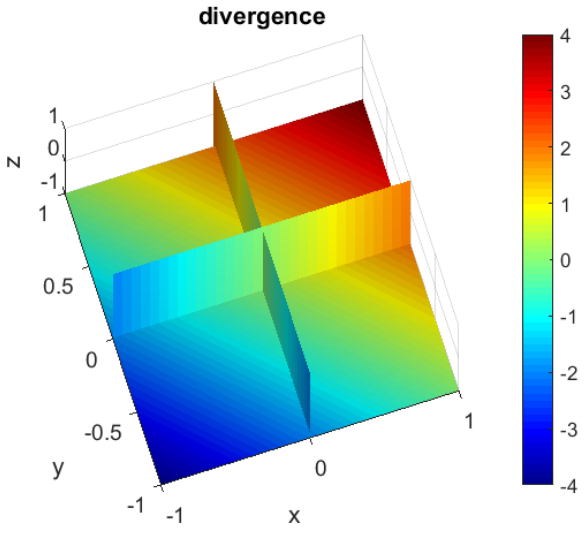


Fig. 10. Vector field and its curl in the XY plane for z = +0.5. **cemDiff04.m**

Fig. 11. [3D] plot of the scalar field for the divergence with slices at x = 0, y = 0 and z = -1.

```
h =
slice(xx,yy,zz,divV,0,0,-1);
```

cemDiff04.m



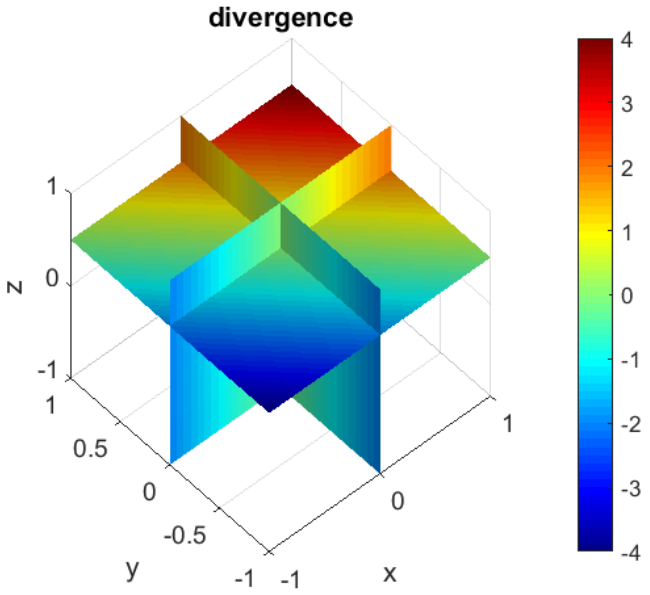


Fig. 11. [3D] plot of the scalar field for the divergence with slices at x = 0, y = 0 and z = +0.5.

```
h = slice(xx,yy,zz,divV,0,0,0.5);
```

cemDiff04.m

The divergence is related to how the vector field changes as you move in the direction of the field. For example, if the vector field points in the x direction and increases in the x direction then the divergence is positive, whereas, if the field points in the x direction and increases in the y direction the divergence would be zero.

LAPLACIAN OPERATOR ∇^2

(9a) $\nabla \equiv \hat{i}\frac{\partial}{\partial x} + \hat{j}\frac{\partial}{\partial z} + \hat{k}\frac{\partial}{\partial z}$ operator called **del**

The **gradient** of a scalar function f is a vector function ∇f .

We can take the divergence of the gradient which gives the **scalar** function called the **Laplacian of *f***

(13) $\nabla \bullet (\nabla f) = \nabla^2 f = \frac{\partial^2 f}{\partial x^2} + \frac{\partial^2 f}{\partial y^2} + \frac{\partial^2 f}{\partial z^2}$ operator called **del2**

The curl of a gradient is **always zero**

(14) $\nabla \times (\nabla f) = (\nabla \times \nabla) f = 0$

The divergence of a curl is **always zero**

(15) $\nabla \bullet (\nabla \times \vec{V}) = 0$

The **Laplacian** of a **vector** can be expressed as

(16) $\nabla^2 \vec{V} = \nabla(\nabla \bullet \vec{V}) - \nabla \times (\nabla \times \vec{V})$

where $\nabla^2 \vec{V} = \hat{i}(\nabla^2 V_x) + \hat{j}(\nabla^2 V_y) + \hat{k}(\nabla^2 V_z)$

The Script **cemDiff05.m** can be used to find and the plot the Laplacian of either a [2D] scalar field $\nabla^2 f(x,y)$ or the Laplacian of a [3D] vector field $\nabla^2 \vec{V}\left(V_x\ V_y\ V_z\right)$. You should download the Script and make sure you understand and can interpret the code.

cemDiff05.m sections of the code

```
% INPUTS
% Number of grid points (integer)    N = 101 default value
           N = 101;
% Range for X, Y and Z values  [minX maxX minY maxY minZ maxZ]
           XY = [-50 50 -50 50 -50 50 ];
         % XY = [-1 1 -1 1 -1 1];
     % flag = 1 for scalar field f or
     % flag = 2 for vector field V(Vx Vy Vz)


     % Define scalar field f
          wLx = 50; wLy = 40;
          kx = 2*pi/wLx; ky = 2*pi/wLy;
          f = sin(kx.*xx) .* cos(ky.*yy);

     % Laplacian
          del2f = del2(f,hx,hy)*4;


     % Define vector field V
          Vxx = xx.^2;
          Vyy = 3 .* xx .* zz.^2;
          Vzz = -2 .* xx .* zz;

% Calculate Laplacian: X Y Z components / magnitude of vector
          del2Vx = del2(Vxx,hx,hy,hz)*6;
          del2Vy = del2(Vyy,hx,hy,hz)*6;
          del2Vz = del2(Vzz,hx,hy,hz)*6;
          del2Vmag = sqrt(del2Vx.^2 + del2Vy.^2 + del2Vz.^2);
```

To calculate the Laplacian in Matlab you use the function **del2**

[2D] `del2f = del2(f,hx,hy)*4;`

[3D] `del2Vx = del2(Vxx,hx,hy,hz)*6;`

where f is the [2D] array for the scalar function; Vxx is the [3D] array for the X component of the vector; and hx, hy and hz are the grid spacing in the X, Y, Z

directions respectively. NOTE: for [2D] **del2** must be multiplied by **4** and for [3D] **del2** is multiplied by **6**.

Figure 12 shows plots of a scalar function $f(x,y)$ and the Laplacian of the function $\nabla^2 f(x,y)$ using the mscript **cemDiff05.m**

$$f(x,y)=\sin\left(\frac{2\pi x}{\lambda_x}\right)\cos\left(\frac{2\pi y}{\lambda_y}\right)$$

$$\nabla^2 f(x,y)=-k_x{}^2\, k_y{}^2\; f(x,y)$$

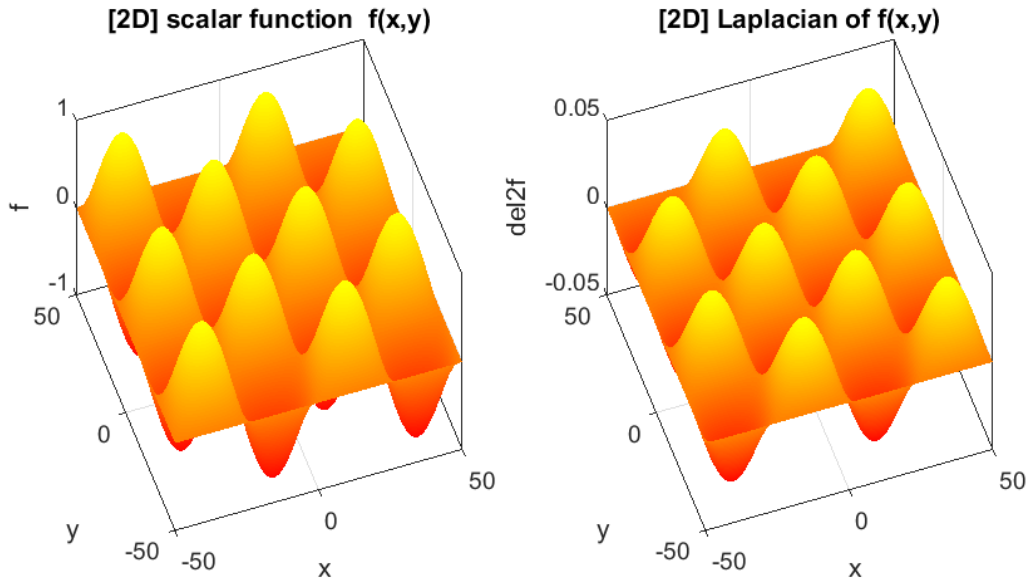


Fig. 12. Surf plots of the function $f(x,y)$ and the Laplacian of the function $\nabla^2 f(x,y)$. **cemDiff05.m**

Figure 13 shows the plots for the Laplacian of the vector function $\vec{V}$ with components

$$V_x=x^2 \quad V_y=3xz^2 \quad V_z=-2xy$$

The Laplacian is given by

$$\nabla^2\vec{V}=2\hat{i}+6x^2\,\hat{j}+0\hat{k} \qquad \left|\nabla^2\vec{V}\right|=\sqrt{4+36x^2}$$

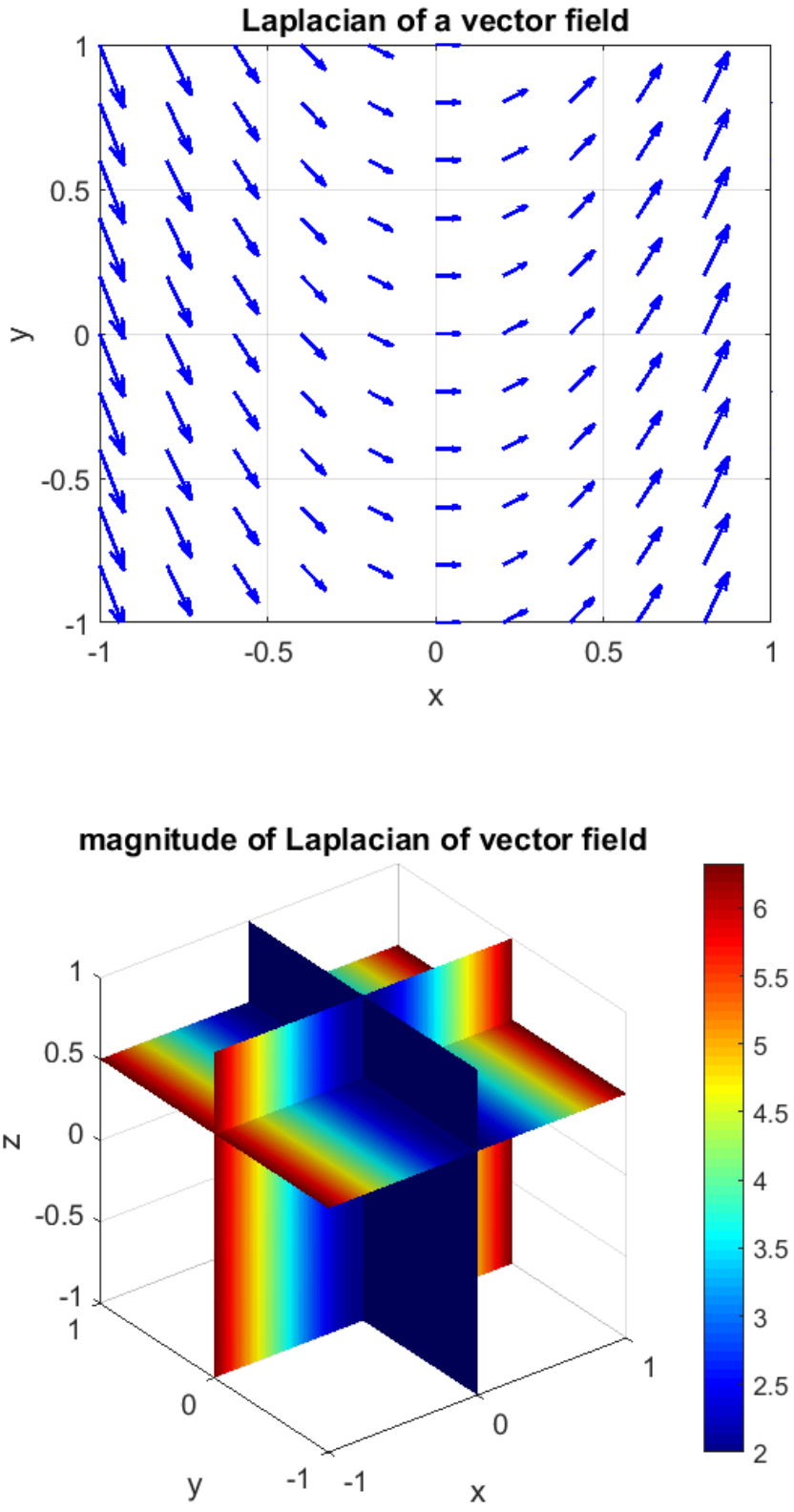


Fig. 13. Laplacian of the vector function $\vec{V}$ with components

$$V_x = x^2 \quad V_y = 3xz^2 \quad V_z = -2xy$$ **cemDiff05.m**