COMP61342: Advanced Computer Vision Module Maths Primer 2013

Carole Twining

Abstract

This document is intended as a very brief overview of the most basic mathematical concepts to be used in this module. The contents should already be familiar to students with a basic maths qualification, but what may be unfamiliar is the range of different notation used for the same mathematical concept in relevant books and papers. Students should read these notes carefully, and make sure they can answer the simple self-test questions.

Contents

1 Differentiation and Differential Calculus

Differential calculus is a area of maths that allows us to describe and analyse the rate of change of quantities. This rate of change can involve the rate of change of a quantity with time, the rate of change of a quantity with spatial position, or in general, the way that any quantity changes as a result of changing the variables on which it depends. It hence in general applies in such areas as considering how the results of a process or algorithm depend on errors on the input data to the process.

1.1 Basic Derivatives

Suppose we have an arbitrary function of a variable x, which we will write as:

$$y = f(x).$$

The slope of the graph of y plotted against x, at the point $x = a$, is given by the derivative:

$$\left.\frac{dy}{dx}\right|_a \equiv \left.\frac{df}{dx}\right|_a .$$

The notation $\cdot|_a$ is often omitted. Where the identity of the independent variable x is clear, the derivative of a function $f(x)$ is sometimes written as $f'(x)$, or just f'.

There are two techniques which allow the calculation of derivatives of more complicated expressions:

- **Product Rule:**
  For a product of functions:

$$f(x) = g(x)h(x) \ \Rightarrow \ f'(x) = g'(x)h(x) + g(x)h'(x).$$

- **Chain Rule:**
  For a function of a function, we have the chain rule:

$$f(x) = g\left(h(x)\right) \ \Rightarrow f'(x) = g'(h(x))h'(x).$$

  Note that some care needs to be taken here in this use of the dash notation, in that $g'(h(x))$ actually means:

$$g'(h(x)) = \left.\frac{dg(h)}{dh}\right|_{h=h(x)},$$

  since the function g only depends indirectly on x via the value of h.

1.1.1 Finite Differences

As we will see below when we consider vectors, we often only know the value of a function $f(x)$ at some discrete sample set of points x, rather than for **any** value of x. In this case, the derivative $f'(x)$ can be approximated using finite-differences. So, if we know the values of $y_i = f(x_i)$ at some set of points $\{x_i : i = 1, 2, \dots n\}$, then the (centred) finite-difference approximation of the derivative is given by:

$$\left.\frac{df(x)}{dx}\right|_{x=x_i} = \left.\frac{dy}{dx}\right|_{x=x_i} \approx \frac{y_{i+1} - y_{i-1}}{x_{i+1} - x_{i-1}} = \frac{f(x_{i+1}) - f(x_{i-1})}{x_{i+1} - x_{i-1}}.$$

If the points x_i are equally-spaced, and a unit distance apart, then this is often written as:

$$\left.\frac{df(x)}{dx}\right|_{x=x_i} \approx \frac{y_{i+1} - y_{i-1}}{2}.$$

Finite-differences are the basis of methods for the numerical solution of equations involving derivatives of functions (*differential equations*). In some cases, you may also encounter the asymmetric difference:

$$\left.\frac{df(t)}{dt}\right|_{t=t_n} \approx \frac{f(t_{n+1}) - f(t_n)}{t_{n+1} - t_n} \quad \text{or:} \quad \left.\frac{df(t)}{dt}\right|_{t=t_n} \approx \frac{f(t_n) - f(t_{n-1})}{t_n - t_{n-1}}.$$

1.2 Partial Derivatives

In most cases, a function of interest will depend on more than one variable. A simple example is the height above sea-level of some part of a landscape, where the exact height depends on our position in two dimensions (which we could think of as latitude and longitude, for example). We hence have a function $f(x, y)$, where we now have two independent variables.

The slope of the landscape at a point obviously depends on which direction we walk in. This information is encoded in the **partial derivatives** (∂ symbol rather than the d we used previously):

$$\left.\frac{\partial f(x,y)}{\partial x}\right|_y, \quad \left.\frac{\partial f(x,y)}{\partial y}\right|_x.$$

What $\left.\frac{\partial f(x,y)}{\partial x}\right|_y$ means is that we take the function $f(x, y)$, and consider only the effect of varying x when we are asking about the slope. We then in effect take the ordinary derivative with respect to x, in that whenever y occurs in the expressions, we treat it as if it were a constant. To give an example:

$$f(x, y) = x^2 y + x^3 + y^2 \ \Rightarrow \frac{\partial f}{\partial x} = 2xy + 3x^2, \ \frac{\partial f}{\partial y} = x^2 + 2y.$$

Note that as above, the $\cdot|_y$ notation is often omitted where the meaning is clear, and in some cases, the even shorter notation is used, where:

$$f_x = \frac{\partial f}{\partial x}.$$

As in the case of the basic derivative, we can also construct finite-difference approximations to these partial derivatives. We can also differentiate these derivatives themselves (so, not just asking what the slope of the landscape is at a point, but how that slope varies with position), and hence we have the **second-order** partial derivatives:

$$\frac{\partial^2 f}{\partial x \partial y} = \frac{\partial}{\partial x}\left(\frac{\partial f}{\partial y}\right) = f_{xy} \equiv f_{yx}.$$

Note that as for the basic derivative, we also have a chain-rule for partial derivatives. So, suppose we have a function f which depends on two independent variables u and v, and that these **both** depend on x and y. That is:

$$f = f(u, v), \;\; u = u(x, y) \;\; \& \;\; v = v(x, y).$$

From before, we can compute what happens if we change **just** u, or **just** v, these are the partial derivatives f_u and f_v. However, if we change **just** x (but **not** y), then both u and v change to some extent since they both depend on x. Hence we have to combine the partial derivatives according to the chain-rule:

$$\left.\frac{\partial}{\partial x}\right|_y f(u, v) = \left.\frac{\partial f}{\partial u}\right|_v \left.\frac{\partial u}{\partial x}\right|_y + \left.\frac{\partial f}{\partial v}\right|_u \left.\frac{\partial v}{\partial x}\right|_y = f_u u_x + f_v v_x.$$

You should take careful note of what variables are being held constant at each step of the computation.

Questions: Differentiation

1. Compute the first derivatives of the two functions $g(x) = x^2$ and $h(x) = \cos x$. Hence:
   - use the product rule to compute the first derivative of $f(x) = x^2 \cos x$.
   - use the chain rule to compute the first derivative of $f(x) = \cos x^2$.

   [Note. Don't let yourself get confused by the fact that x is used in the definition of both g and h.]

2. Consider the function $f(x, y) = x^2 + y^2$. Find by direct computation the partial derivatives $f_x, f_y, f_{xx}, f_{xy}, f_{yx}$ and f_{yy}.

   Consider now the function $g(x, y) = (x+y)^2 + (x-y)^2$. By direct computation, find the first partial derivatives g_x and g_y.

   Now consider the substitution $u(x, y) = x + y$, and $v(x, y) = x - y$. Hence by use of the chain rule for partial derivatives, express the first partial derivatives of g in terms of the first partial derivatives of f, and verify that you get the same answer as you did by direct computation.

2 Arrays, Vectors, and Matrices

2.1 Data Arrays

For a multi-valued data set (for example, the daily average temperature at a location, measured every day for a year), the entire set of values is often combined into a single structure. So, we could arrange the temperature data described above into a vector $\vec{T}$ with 365 elements, where the i^{th} element T_i was the temperature on the i^{th} day.

The notation used for vectors varies greatly, so you will find $\vec{T}$, $\underline{\mathrm{T}}$, $\mathbf{T}$, and sometimes just T all used to denote the entire vector of temperatures. How you refer to an individual element also varies, so you will find T_i and also $T(i)$ (and possibly $T(1, i)$ and $T(i, 1)$) all used in various mathematical and coding contexts.

Now suppose that we measure the daily temperature not just at one location, but at some set of locations. We can then form a two-dimensional array of data values. For example, we could have the temperature on the i^{th} day at location α, which we could refer to as the element $T_{i\alpha}$ of the array $\mathbf{T}$. We could also describe this as the element $T(i, \alpha)$. In Matlab notation, the day i is the row index, and the location α is the column index. This data array is rectangular and complete $-$ we have measurements at *all* locations for *every* day.

Note that we have been careful here, and used different alphabets to denote the day index i and the location index α. Not all authors are this careful, and for instance, you will find instances where authors will refer to an element T_{ij}, where i denotes day (and runs from 1 to 365), whereas j denotes location (and runs from 1 to 7, say). It is then important to note that T_{ji} is *different* to T_{ij} (the temperature at location one on the seventh day in general will have nothing to do with the temperature at the seventh location on the first day), and may not even exist. Hence care needs to be taken when referring to individual elements of an array as to which way the array has been organized, and what the indices actually refer to.

We can also construct multi-dimensional arrays. Suppose that at each location for each day, we also measured the temperature at ground level, and at a height of 10 metres above the ground. We could then form these into a three-dimensional array, with elements $T_{i\alpha A}$, where i is the day index, α is the location index, and A the height index. In Matlab notation, we could refer to element $T(355, 7, 2)$, which would be day 355, at the seventh location, and the temperature at the height.

2.2 Vectors and Operations on Vectors

Now lets look just at vectors. In mathematical terms, we just have a vector $\vec{x}$ with N elements x_i. In coding terms, you have to distinguish between **column vectors**, which you can think of as an array with elements $\{x(i,1) : i = 1, 2, \ldots N\}$, and a **row vector** $\{y(1,i) : i = 1, 2, \ldots N\}$. If $\vec{y}$ is a row(column) vector, the corresponding column(row) vector is given by the **transpose**, $\vec{y}^T$, where $y^T(1,i) = y(i,1)$ and so on.

Some Basic Operations with Vectors

- **Multiplication by a Number:**
  For a vector $\vec{x}$ and a number a, we form the new vector $a\vec{x}$ with elements ax_i.

- **Addition and Subtraction:**
  If $\vec{x}$ and $\vec{y}$ are both vectors of the same size and type (i.e., both row vectors or both column vectors), then we can add them, to obtain a new vector:

  $$\vec{z} = \vec{x} + \vec{y},\ \ z_i = x_i + y_i,\ \ z(i,1) = x(i,1) + y(i,1)\ \text{ or: }\ z(1,i) = x(1,i) + y(1,i).$$

  If we have a column vector and a row vector of the same length, in coding terms we can still add them as $\vec{z} = \vec{x} + \vec{y}^T$. As before, in mathematical terms, we do not usually distinguish between row and column vectors.

  Since we can add vectors, we can obviously subtract them as well, by just rearranging the above expressions.

- **Dot Product:**
  If we have two vectors of the same size N, $\vec{x}$ and $\vec{y}$, the **dot product** is the number given by:

  $$\vec{x} \cdot \vec{y} = \sum_{i=1}^{N} x_i y_i.$$

  If they are both row vectors, in coding notation this is written as $x^T y$ (which is the *same* as $y^T x$), and if they are both column vectors, as xy^T (or equivalently yx^T).

  Length/Modulus of a Vector: [1]
  In terms of geometry, we can think of the dot product of a vector with itself, $\vec{x} \cdot \vec{x}$, as the length of the vector *squared*, which is sometimes written as $|\vec{x}|^2$.

  Angle Between Vectors:
  The dot product of two vectors, $\vec{x} \cdot \vec{y}$ can then be thought of as the number $|\vec{x}|\ |\vec{y}| \cos\theta$, where θ is defined as the angle between the two vectors.

  Two vectors are said to be **orthogonal** if the angle between them is ninety degrees, hence their dot product is zero.

- **Outer Product:**
  If we have two vectors $\vec{v}$ and $\vec{w}$, which need not be of the same size, we can form a matrix by taking the outer product[2] of $\vec{v}$ and $\vec{w}$:

  $$\mathbf{U} = \vec{v} \otimes \vec{w},\ \ U_{ij} = v_i w_j.$$

  We hence see that the first vector uses the row index, and the second uses the column index. So, if $\vec{v}$ is of size m, and $\vec{w}$ is of size n, then $\vec{v} \otimes \vec{w}$ is a matrix of size $m \times n$. In MATLAB, if $\vec{v}$ and $\vec{w}$ are written as row vectors, then the outer product is given by: `v'*w` where `v'` is the column vector formed by taking the transpose of `v`. We hence see that this corresponds to a (single-column) matrix `v'` of size $m \times 1$ multiplying `w` of size $1 \times n$ (a single-row matrix) to produce a matrix of size $m \times n$.

[1] You should not confuse the length of a vector as used in MATLAB, where it means just the number of elements, with the length $|\vec{x}|$ as defined here. In MATLAB, what I here refer to as the length is given by **norm(vector,2)**, which returns the square root of the sum-of-the-squares of the elements of **vector**.

[2] Note that the definition of the outer product given in the old version of the Maths Primer by Neil Thacker, that you might have come across on the TINA website disagrees with this definition. But the one used here is in accord with the definition of the tensor product.

- **The Cross Product**
  For the special case where we have 2 vectors in three dimensions $\vec{v} = (v_x, v_y, v_z)$, $\vec{w} = (w_x, w_y, w_z)$, we can take the cross-product, to give another vector in three dimensions. The physical meaning of this vector is that it is at right angles to both of the original vectors, and its length is given by $|\vec{v}||\vec{w}|\sin\theta$, where θ is the angle between the two vectors (see *angle between vectors* above). The notation for the cross-product (sometimes also called the *wedge product*) is $\vec{v} \times \vec{w}$ or $\vec{v} \wedge \vec{w}$. Care should be taken not to confuse this with the outer product! In components, the cross-product is given by:

  $$\vec{v} \times \vec{w} = (v_y w_z - v_z w_y, v_z w_x - v_x w_z, v_x w_y - v_y w_x).$$

  We hence see that the x component involves the y and z components of the two vectors, and similarly for the other components. To remember the signs, think of xyz, block out the component you are trying to find, and take remaining indices. So, y component of the cross-product gives xz, which means takes x component of $\vec{v}$ and z component of $\vec{w}$ with plus sign, and z component of $\vec{v}$ and x component of $\vec{w}$ with a minus sign.

  The appearance of the minus signs means that $\vec{v} \times \vec{w} = -\vec{w} \times \vec{v}$.

2.2.1 Matrices Acting on Vectors

Using the dot product above, we can let a (row) vector $\vec{y}$ act on a (column) vector $\vec{x}$, written as $\vec{y} \cdot \vec{x}$, $y^T x$, or $\sum_i y_i x_i$.

Now suppose that rather than a *single* vector $\vec{y}$, we have a collection of row vectors, and we wish to know the result of taking the dot product between each of these vectors in turn with the vector $\vec{x}$. If $\vec{y}_\alpha$ is the α^{th} row vector, with elements $y_{\alpha i}$, then the dot product for this vector is given by $\vec{y}_\alpha \cdot \vec{x} = \sum_i y_{\alpha i} x_i$.

We can also compute this by stacking the row vectors into a matrix, $\mathbf{Y}$, where the α^{th} row of this matrix is the vector $\vec{y}_\alpha$. The collection of dot products with $\vec{x}$ can then be written as the product of the matrix and the vector $\mathbf{Y}\vec{x}$. The result is another column vector, where:

$$\vec{z} = \mathbf{Y}\vec{x}, \;\; \vec{z}_\alpha = \sum_i Y_{\alpha i} x_i = \sum_i y_{\alpha i} x_i.$$

We can then see that if $\vec{x}$ is a column vector with N elements, and $\mathbf{Y}$ is a matrix with M rows (and of course N columns), then the result of the multiplication is a column vector with M elements.

Example: Rotation in the plane.

Take a point in the plane with coordinates (x, y). If we rotate the point through an angle θ about the origin, the coordinates of the rotated point are given by:

$$x \to x\cos\theta + y\sin\theta, \;\; y \to y\cos\theta - x\sin\theta.$$

We now collect the coordinates into a column vector $\vec{r}$, with components $r_1 = x$ and $r_2 = y$. We can then write this rotation in matrix form as:

$$\vec{r} \to \mathbf{R}(\theta)\vec{r}, \;\; \textit{where: } \mathbf{R}(\theta) = \begin{pmatrix} \cos\theta & \sin\theta \\ -\sin\theta & \cos\theta \end{pmatrix}.$$

Suppose we now have a rotation followed by a translation, so that:

$$x \to x\cos\theta + y\sin\theta + a, \;\; y \to y\cos\theta - x\sin\theta + b.$$

One way to write this would be to use $\mathbf{R}(\theta)\vec{r} + \vec{t}$, where $\vec{t}$ is the translation vector with elements a and b. However, we can write this combined rotation and translation in terms of a single matrix. We extend the vector $\vec{r}$, where $r_1 = x$, $r_2 = y$, and $r_3 = 1$. Using this 3-element vector, we can now write:

$$\vec{r} \to \mathbf{R}(\theta, a, b)\vec{r}, \;\; \textit{where: } \mathbf{R}(\theta, a, b) = \begin{pmatrix} \cos\theta & \sin\theta & a \\ -\sin\theta & \cos\theta & b \\ 0 & 0 & 1 \end{pmatrix}.$$

If you perform the matrix multiplication, you will see that the formula works. The point of this is that by extending $\vec{r}$ in this way, we have reduced a translation and rotation (a matrix multiplication followed by a vector addition), to one matrix multiplication, and this is a form that is often used in applications.

2.3 Matrices and Matrix Algebra

We have seen how a (column) vector of size N can be multiplied by a matrix of size $M \times N$ (M rows and N columns) to produce as output a vector of size M. We now consider the matrix acting on a set of P such column vectors, rather than a single column vector. So, if we have a column vector $\vec{x}_A$ with elements x_{iA}, multiplied by the matrix $\mathbf{Y}$ as before, we obtain:

$$\vec{z}_A = \mathbf{Y}\vec{x}_A \;\; \Rightarrow \;\; \mathbf{Z} = \mathbf{YX}, \text{ where: } X_{iA} = x_{iA},$$

where $\mathbf{Y}$ is of size $M \times N$, the collection of P column vectors form a matrix $\mathbf{X}$ of size $N \times P$, and the result of the multiplication is a matrix $\mathbf{Z}$ of size $M \times P$, with elements[3]:

$$Z_{\alpha A} = \sum_i Y_{\alpha i} X_{iA}.$$

We now summarize the basic rules for matrix algebra.

Some Basic Operations with Matrices

- **Transpose:**
  If $\mathbf{A}$ is matrix with M rows and N columns, the transpose $\mathbf{A}^T$ is a matrix formed by exchanging rows and columns, hence is a matrix with N rows and M columns, with elements:

  $$(\mathbf{A}^T)_{\alpha i} = A_{i\alpha}.$$

- **Addition:**
  If matrices $\mathbf{A}$ and $\mathbf{B}$ are the same size, then we can add them by adding the corresponding elements:

  $$\mathbf{C} = \mathbf{A} + \mathbf{B} \text{ where: } C_{i\alpha} = A_{i\alpha} + B_{i\alpha}.$$

- **Multiplication:**
  As we have already seen, if we have a matrix $\mathbf{A}$ of size $M \times N$ and a matrix $\mathbf{B}$ of size $N \times P$, then we can form the product:

  $$\mathbf{C} = \mathbf{AB}, \;\; C_{\alpha A} = \sum_i A_{\alpha i} B_{iA}.$$

  It is important to note that unless we have square matrices (same number of rows as columns), even if $\mathbf{AB}$ exists, $\mathbf{BA}$ does not. And even for square matrices, $\mathbf{AB}$ is not in general the same as $\mathbf{BA}$. Hence order of multiplication matters for matrices.

- **Identity:**
  The Identity matrix $\mathbf{I}$ is a square matrix of size $N \times N$, with all the diagonal elements $I_{jj} = 1$, and zeros everywhere else. It has the property that:

  $$\mathbf{IA} = \mathbf{A}, \;\; \mathbf{BI} = \mathbf{B}$$

  where $\mathbf{A}$ is *any* matrix of size $N \times P$, and $\mathbf{B}$ is any matrix of size $P \times N$ (and for any value of $P \geq 1$).

- **Inverse:**
  If a matrix $\mathbf{A}$ has an inverse $\mathbf{A}^{-1}$, then this inverse has the property that:

  $$\mathbf{A}^{-1}\mathbf{A} = \mathbf{I} \text{ and } \mathbf{AA}^{-1} = \mathbf{I}.$$

  Such an inverse does not always exist, but where it does, it takes the place in matrix algebra of division in ordinary algebra.

[3]Note that we have again been careful with our notation, and used different alphabets for indices which have different ranges. This is not always the case, and you will often come across expressions such as:

$$Z_{jk} = \sum_i Y_{ji} X_{ik},$$

where the use of the same alphabet does not necessarily mean that each index has the same range.

2.3.1 Eigenvectors and Eigenvalues

We've already mentioned matrices acting on vectors. There is a special case of a matrix acting on a vector, which occurs frequently in many fields of computing and physics. This is the concept of an **eigenvector**. For a square matrix $\mathbf{A}$, a (non-zero) vector $\vec{v}$ is an eigenvector of $\mathbf{A}$ if it satisfies:

$$\mathbf{A}\vec{v} = \lambda\vec{v},$$

where λ is some number (the **eigenvalue** associated with the eigenvector $\vec{v}$).

What this means is that the matrix $\mathbf{A}$ acts on the vector $\vec{v}$ to produce another vector, which is just some multiple of the original vector. Hence applying $\mathbf{A}$ changes the length/modulus of the vector, but *not* its direction.

MATLAB has standard functions for computing the eigenvectors and eigenvalues of a matrix.

Questions: Vectors and Matrices

1. Consider the vectors $\vec{a} = (0, 1, 3, 4)$, and $\vec{b} = (0, 3, 4, 1)$.
   - Compute the dot product of these vectors.
   - Compute $\mathbf{C}$, the outer product $\vec{a} \otimes \vec{b}$.
   - Compute $\mathbf{C}\vec{b}$, and see how this can be related to $\vec{a}$. Look at the formula for the outer product, to see if this is the result you might have expected (Hint: Try working out $\vec{b} \cdot \vec{b}$. Does this give you any ideas?).
   - If I have the equation $\mathbf{AB} = \mathbf{X}$, how do I write the form of the solution for $\mathbf{B}$ if I know $\mathbf{A}$ and $\mathbf{X}$? (Hint: What would you do if these were numbers, rather than matrices? Does this give you any idea as to which matrix operation you require, the analog of what you did with numbers?)
2. Consider the vectors $\vec{v} = (3, 4, 0)$ and $\vec{w} = (-4, 3, 0)$.
   - Compute the cross-product $\vec{c} = \vec{v} \times \vec{w}$
   - By computing the dot-products $\vec{c} \cdot \vec{v}$ and $\vec{c} \cdot \vec{w}$, show that $\vec{c}$ is at right angles to both vectors.
   - By computing $|\vec{c}|$, $|\vec{v}|$ and $|\vec{w}|$, work out the angle between $\vec{v}$ and $\vec{w}$. Does this agree with the result of $\vec{v} \cdot \vec{w}$?

3 Vector Calculus

In §1, we considered differentiating (scalar) functions, and in §2, we considered vectors and arrays. We can now put these together, and consider differentiating vector-valued functions.

Which might seem a rather odd thing to want to do, so let's give a simple example.

Suppose we have a fluid flowing through a pipe, or just tea in our tea-cup after stirring. We then have positions in the fluid, relative to the cup or pipe, which we will write as a position in three-dimensional space using the position vector $\vec{r} = (x, y, z)$, where we have used the usual Cartesian xyz coordinates.

At any instant t, at any point in the fluid, we then have the velocity of the fluid at that point, which we will call $\vec{v}(x, y, z, t)$ or $\vec{v}(\vec{r}, t)$. This velocity is obviously a vector (which direction is the fluid moving in, and how fast?), it depends on our position $\vec{r}$, hence it is a vector-valued function of another vector.

When it comes to the pattern of flow in the fluid, we can consider two sorts of variation; how the flow varies over time (that is, we have $\vec{v}(\vec{r}, t)$, where t is time), and how the flow varies with position. For computing how the flow changes in space and in time, we have the partial derivatives we introduced earlier: $\frac{\partial}{\partial t}$, $\frac{\partial}{\partial x}$, $\frac{\partial}{\partial y}$, and $\frac{\partial}{\partial z}$. It is usual to take the spatial partial derivatives and collect then together into the *del* operator:

$$\vec{\nabla} = \left(\frac{\partial}{\partial x}, \frac{\partial}{\partial y}, \frac{\partial}{\partial z}\right).$$

You can think of this a bit like a vector, hence acting on a scalar (such as the temperature of the tea!), it should give a vector. Acting on a vector, we can think of it as dot-product like, to give a scalar, or cross-product like to give another vector. This then gives us the following first-order and second-order differential operators:

- **First Derivatives:**
  - Gradient of a *scalar* function $f(\vec{r})$ giving a vector:

$$\vec{\nabla} f(\vec{r}) = \left(\frac{\partial f(\vec{r})}{\partial x}, \frac{\partial f(\vec{r})}{\partial y}, \frac{\partial f(\vec{r})}{\partial z}\right) = (f_x, f_y, f_z)$$

- Applied to a *vector-valued* function:
  1. **Divergence**, giving a scalar[4]:
  $$\vec{\nabla} \cdot \vec{v}(\vec{r}) = \frac{\partial v_x(\vec{r})}{\partial x} + \frac{\partial v_y(\vec{r})}{\partial y} + \frac{\partial v_z(\vec{r})}{\partial z.}$$
  2. **Curl**, giving a vector:
  $$\vec{\nabla} \times \vec{v}(\vec{r}) = (v_{zy} - v_{yz}, v_{xz} - v_{zx}, v_{yx} - v_{xy}),$$
  where we have used the shorthand:
  $$v_{yx} = (v_y)_x = \frac{\partial v_y}{\partial x},$$
  hence the seeming flip of the indices in the cross-product.

- **Second Derivatives:**
  - Of a scalar function (*div-grad* or *del-squared*, Δ is an alternative notation for ∇^2, also called the *Laplace operator*):
  $$\Delta f(\vec{r}) = \nabla^2 f(\vec{r}) = \vec{\nabla} \cdot \left(\vec{\nabla} f(\vec{r})\right) = f_{xx} + f_{yy} + f_{zz} = \frac{\partial^2 f(\vec{r})}{\partial x^2} + \ldots.$$
  - Of a vector-valued function (the *vector Laplacian*, *grad-div* minus *curl-curl*):
  $$\vec{\nabla}^2 \vec{v}(\vec{r}) = \vec{\nabla}(\vec{\nabla} \cdot \vec{v}) - \vec{\nabla} \times (\vec{\nabla} \times \vec{v}) = \left(\vec{\nabla}^2 v_x, \vec{\nabla}^2 v_y, \vec{\nabla}^2 v_z\right).$$

3.1 Discrete Laplacian

We have seen what the Laplacian is in the continuum, but what does it look like if we are using finite differences?

So, let us suppose we have the values of our function at a set of points forming a two-dimensional regular grid, $f(x_i, y_j)$, or simple $f(i, j)$, where the integers i and j give us the position on the grid. We will assume the grid has unit spacing.

Near a point (i, j), we can construct two finite-difference versions of f_x:

$$f_x(i + \frac{1}{2}, j) \approx f(i+1, j) - f(i, j), \;\; f_x(i - \frac{1}{2}, j) \approx f(i, j) - f(i-1, j).$$

Taking the difference of these gives a finite-difference estimate of $f_{xx}(i, j)$:

$$f_{xx}(i, j) \approx f(i+1, j) + f(i-1, j) - 2f(i, j).$$

Similarly:

$$f_{yy}(i, j) \approx f(i, j+1) + f(i, j-1) - 2f(i, j).$$

We hence have our first version of the discrete Laplacian, the sum of $f_{xx}(i, j)$ and $f_{yy}(i, j)$, which can be written in a compact form as the matrix of numbers:

$$D^2 = \begin{bmatrix} 0 & 1 & 0 \\ 1 & -4 & 1 \\ 0 & 1 & 0 \end{bmatrix},$$

where the centre of the matrix corresponds to the coefficient for $f(i, j)$, and the other values correspond to the coefficients of the terms at the neighbouring sites. Note that the approximation to a derivative means that the sum of the coefficients has to be zero.

This form is useful, but doesn't use all the information from the nearest-neighbours of (i, j) to compute the approximation, and leaves out the diagonal neighbours. We can include these, and by using the zero-sum rule, this gives the second version of the discrete Laplacian:

$$\tilde{D}^2 = \begin{bmatrix} 1 & 1 & 1 \\ 1 & -8 & 1 \\ 1 & 1 & 1 \end{bmatrix}$$

[4] Note the dual use of the sub-scripts compared with the gradient, and don't confuse the subscript in v_x meaning x-component, with the subscript in f_x meaning partial derivative with respect to x.