

MAA Oral

Victor Telea

Contents

1	Integration techniques	2
2	Ordinary differential equations (ODEs)	2
3	Linear algebra	3
4	Graph Theory	4

1 Integration techniques

For this topic, this trigonometric integral is essential: $\int \frac{dx}{x^2+1} = \arctan(x) + C$

When we have the **integral of a rational function** (i.e., a function of the form $\frac{f(x)}{g(x)}$), we begin by considering the degrees of the polynomials in the numerator and denominator. If $\deg(f(x)) \geq \deg(g(x))$ (i.e., the rational function is *improper*), we first apply **Euclidean division** to rewrite it as a polynomial plus a proper rational function.

Once a proper fraction remains, we apply the method of **partial fractions**. To do this, we factor the denominator $g(x)$, which we can express in general as:

$$g(x) = c \prod_{i=1}^k (x + r_i)^{m_i} \prod_{j=1}^l (x^2 + p_j x + q_j)^{n_j}$$

Here, c is a constant; the first product runs over the real roots of $g(x)$, and the second over the irreducible quadratic factors (corresponding to complex roots).

It follows that we can always decompose $\frac{f(x)}{g(x)}$ as:

$$\frac{f(x)}{g(x)} = \sum_{i=1}^k \sum_{s=1}^{m_i} \frac{A_{i,s}}{(x + r_i)^s} + \sum_{j=1}^l \sum_{t=1}^{n_j} \frac{B_{j,t}x + C_{j,t}}{(x^2 + p_j x + q_j)^t}$$

Where we go through all roots of $g(x)$, and each one of their multiplicities.

(This applies to all rational functions since every real polynomial factors uniquely into linear and irreducible quadratic terms by the fundamental theorem of algebra.)

2 Ordinary differential equations (ODEs)

We will go through the different types of differential equations and their respective method of solving. DEs can be categorised by:

- Ordinary vs Partial (Always ordinary)
- Linear vs non-Linear (Categorisation of the coefficients)
- Heterogeneous vs Homogeneous (Whether there is a term without y or one of its derivatives)
- Their order (Highest derivative of y)

Separable ODEs are of the form $y' = f(x)g(y)$.

These can be solved by the following manipulation:

$$\begin{aligned} y' &= f(x)g(y) \\ \Leftrightarrow \frac{dy}{dx} &= f(x)g(y) \\ \Leftrightarrow \frac{1}{g(y)} dy &= f(x) dx \\ \Leftrightarrow \int \frac{1}{g(y)} dy &= \int f(x) dx \end{aligned}$$

Linear ODEs are of the form $y' + f(x)y = g(x)$.

For these we find y_h (homogenous solution, $g(x)=0$) first, and then we find the y_p (particular solution). It follows that $y_h + y_p = y_g$ (general solution.) The homogenous solution can be found by the separable method.

For the particular solution we use we use variation of constants. We have only seen this for three types of functions, one must choose the right terms to match the inhomogenous term.

- Polynomial: x^n : use $a_0 + \dots + a_n x^n$
- Exponential: e^{mx} : use Ae^{mx}
- Trigonometric: $\sin(mx), \cos(mx)$: use $A \cos(mx) + B \sin(mx)$
- In case of overlap with homogeneous solution: multiply by x

Then we input these into the ODE, and solve the system of equations that results.

Linear ODEs can also be solved by the **Integration Factor Method** which is a direct formula. This can be helpful, but in some cases the previous method is better as integration by parts can appear here. We have $\mu(x) = e^{\int f(x) dx}$ as the integration factor. The formula to solve $y' + f(x)y = g(x)$ is:

$$y = e^{-\int f(x) dx} \left(\int g(x) \cdot e^{\int f(x) dx} dx + C \right)$$

ODEs of order 2. We've only seen them in the form $ay'' + by' + cy = f(x)$. There are 2 methods to solve these.

Using the axillary equation $ax^2 + bx + c = 0$ we can find y_h .

- 2 distinct real roots (m_1, m_2): $y_h = Ae^{m_1 x} + Be^{m_2 x}$
- 1 real repeated root (m_1): $y_h = Ae^{m_1 x} + Bxe^{m_1 x}$
- Complex roots ($p \pm \omega i$): $y_h = e^{px}(A \cos(\omega x) + B \sin(\omega x))$

We can also reduce the ODE to a first-order system: The ODE can be reduced by introducing a substitution to express it as two first-order equations.

Set $z = ay' + \alpha y$ for a suitable constant α (often chosen to simplify the resulting equations). Then the original equation becomes a system:

$$\begin{cases} z = ay' + \alpha y \\ z' + \beta z = g(x) \end{cases}$$

for some constant β , determined by matching coefficients in the original ODE. This reduction allows solving sequentially: first the equation for z , then recovering y from the first equation.

For y_p for ODEs of order 2, we apply variation of constants analogously as in linear ODEs. Here we also need to be careful in case of overlap.

3 Linear algebra

Let $M_{i,j}$, a minor of A , be a matrix of size $(n-1) \times (n-1)$ where we remove the union of row i and column j .

Let $C_{i,j}$ be the cofactor of $M_{i,j}$, s.t. $C_{i,j} = (-1)^{i+j} |M_{i,j}|$.

Let the cofactor matrix of A be the matrix composed of all such cofactors $C_{i,j}$, arranged in the same positions as the entries of A :

$$\text{CofactorMatrix}(A) = \begin{pmatrix} C_{1,1} & C_{1,2} & \cdots & C_{1,n} \\ C_{2,1} & C_{2,2} & \cdots & C_{2,n} \\ \vdots & \vdots & \ddots & \vdots \\ C_{n,1} & C_{n,2} & \cdots & C_{n,n} \end{pmatrix}$$

Laplace expansion: $|A| = \sum_{i=1}^n (-1)^{i+j} a_{i,j} |M_{i,j}| \quad \forall j$

(We can have the sum in function of either i or j , but the other variable is fixed)

Properties

- $|A| = |A^T|$
- $|A| \xrightarrow{R_{ij}} |B| \implies |B| = -|A|$ (row swap changes sign)
- $|A| \xrightarrow{kR_i} |B| \implies |B| = k|A|$ (scaling a row by k)
- $|R'_1 + R''_1, R_2, \dots, R_n| = |R'_1, R_2, \dots, R_n| + |R''_1, R_2, \dots, R_n|$ (linearity in a row)
- $|A| \xrightarrow{R_i + kR_j} |B| \implies |B| = |A|$ (adding multiple of one row to another does not change determinant)
- $|A \cdot B| = |A| \cdot |B|$

All of these properties apply to rows, due to the fact that i, j are interchangeable in the laplace expansion, all of these apply for columns aswell.

Inverse of a non-singular square matrix ($|A| \neq 0$): $A^{-1} = \frac{\text{Adj}(A)}{|A|}$ where $\text{Adj}(A)$ is the transpose of the cofactor matrix.

Cramer's Rule. Given a system $AX = B$, where $A \in \mathbb{R}^{n \times n}$ is invertible and $X, B \in \mathbb{R}^{n \times 1}$, the solution is given by:

$$x_i = \frac{|A_i|}{|A|}, \quad \text{where } A_i \text{ is } A \text{ with its } i\text{-th column replaced by } B.$$

Change of basis (matrix representation): Let V and W be finite-dimensional vector spaces, and let $T : V \rightarrow W$ be a linear transformation. Given a basis B for V and a basis C for W , the matrix representation of T with respect to these bases is

$$[T]_C^B = Q^{-1}TP$$

where P is the change-of-basis matrix from the standard basis of V to B , and Q is the change-of-basis matrix from the standard basis of W to C .

4 Graph Theory

To find the **Minimum Spanning Tree** (MST) one can use one of two algorithms:

Prim's Algorithm: Start from any vertex and repeatedly add the smallest-weight edge that connects a visited vertex to an unvisited one, until all vertices are included.

Kruskal's Algorithm: Sort all edges by weight and add them one by one, skipping any edge that would form a cycle, until all vertices are included.

Prim's algorithm can also be applied to the adjacency matrix. Start by selecting any vertex, then choose the smallest edge weight in its row. Mark the selected vertex as included, and repeat the process by examining the row corresponding to the newly selected vertex. Continue until all vertices are included.

Traveling Salesman Problem (TSP) otherwise known as: Minimum Weight Hamiltonian cycle. The first step is to make the graph complete. There is no known algorithm to solve it directly, but there are strategies for a lower and upper bound:

The **lower bound** is found using the **deleted vertex algorithm**: for each vertex, delete it from the graph, find the **MST** of the remaining vertices, then add the weights of the two shortest edges connecting the deleted vertex back to the **MST**. Repeat this for all vertices and take the maximum of the resulting values.

The **upper bound** is found using the **nearest neighbour algorithm**: for each starting vertex, construct a tour by repeatedly visiting the nearest unvisited vertex until all have been visited, then return to the starting point. Compute the total weight of each such tour and take the minimum of these values.

Dijkstra's algorithm can be applied using a table. Initialize distances from the start vertex (zero for start, infinity for others). At each step, select the unvisited vertex with the smallest current distance, update the distances to its neighbors by checking edge weights in the table, and mark the vertex as visited. Repeat until all vertices are processed or shortest paths are found. (Note the distances as $A/0$, $B/5$ or C/∞).