Appendix A

Useful identities and theorems from vector calculus

A.1 Vector identities

$$\mathbf{A}\cdot(\mathbf{B}\times\mathbf{C}) = \mathbf{C}\cdot(\mathbf{A}\times\mathbf{B}) = \mathbf{B}\cdot(\mathbf{C}\times\mathbf{A})$$

$$\mathbf{A}\times(\mathbf{B}\times\mathbf{C}) = \mathbf{B}(\mathbf{A}\cdot\mathbf{C}) - \mathbf{C}(\mathbf{A}\cdot\mathbf{B})$$

$$(\mathbf{A}\times\mathbf{B})\times\mathbf{C} = \mathbf{B}(\mathbf{A}\cdot\mathbf{C}) - \mathbf{A}(\mathbf{B}\cdot\mathbf{C})$$

$$\nabla\times\nabla f = 0$$

$$\nabla\cdot(\nabla\times\mathbf{A}) = 0$$

$$\nabla\cdot(f\mathbf{A}) = (\nabla f)\cdot\mathbf{A} + f(\nabla\cdot\mathbf{A})$$

$$\nabla\times(f\mathbf{A}) = (\nabla f)\times\mathbf{A} + f(\nabla\times\mathbf{A})$$

$$\nabla\cdot(\mathbf{A}\times\mathbf{B}) = \mathbf{B}\cdot(\nabla\times\mathbf{A}) - \mathbf{A}\cdot(\nabla\times\mathbf{B})$$

$$\nabla(\mathbf{A}\cdot\mathbf{B}) = (\mathbf{B}\cdot\nabla)\mathbf{A} + (\mathbf{A}\cdot\nabla)\mathbf{B} + \mathbf{B}\times(\nabla\times\mathbf{A}) + \mathbf{A}\times(\nabla\times\mathbf{B})$$

$$\nabla\cdot(\mathbf{A}\,\mathbf{B}) = (\mathbf{A}\cdot\nabla)\mathbf{B} + (\mathbf{B}\cdot\nabla)\mathbf{A}$$

$$\nabla\times(\mathbf{A}\times\mathbf{B}) = (\mathbf{B}\cdot\nabla)\mathbf{A} - (\mathbf{A}\cdot\nabla)\mathbf{B} - \mathbf{B}(\nabla\cdot\mathbf{A}) + \mathbf{A}(\nabla\cdot\mathbf{B})$$

$$\nabla\times(\nabla\times\mathbf{A}) = \nabla(\nabla\cdot\mathbf{A}) - \nabla^2\mathbf{A}$$

A.2 The gradient theorem

For two points $\mathbf{a}$, $\mathbf{b}$ in a space where a scalar function f with spatial derivatives everywhere well-defined up to first order,

$$\int_{\mathbf{a}}^{\mathbf{b}} (\nabla f) \cdot \mathrm{d}\boldsymbol{\ell} = f(\mathbf{b}) - f(\mathbf{a}) \ ,$$

independently of the integration path between $\mathbf{a}$ and $\mathbf{b}$.

A.3 The divergence theorem

For any vector field $\mathbf{A}$ with spatial derivatives of all, its scalar components everywhere well-defined up to first order,

$$\int_V (\nabla \cdot \mathbf{A}) \mathrm{d}V = \oint_S \mathbf{A} \cdot \hat{\mathbf{n}} \, \mathrm{d}S \ ,$$

where the surface S encloses the volume V.

A.4 Stokes' theorem

For any vector field $\mathbf{A}$ with spatial derivatives of all, its scalar components everywhere well-defined up to first order,

$$\int_S (\nabla \times \mathbf{A}) \cdot \hat{\mathbf{n}} \, \mathrm{d}S = \oint_\gamma \mathbf{A} \cdot \mathrm{d}\boldsymbol{\ell} \ ,$$

where the contour γ delimits the surface S, and the orientation of the unit normal vector $\hat{\mathbf{n}}$ and direction of contour integration are mutually linked by the right-hand rule.

Appendix B

Coordinate systems and the fluid equations

This Appendix is adapted in part from Appendix B of the book by Jean-Louis Tassoul entitled *Theory of Rotating Stars* (Princeton University Press, 1978), with a number of additions, including the MHD induction equation, expressions for the operators $\mathbf{u}\cdot\nabla$, $\nabla\times\nabla\times$, and ∇^2 acting on a vector field. Also, the Note that in sections B.1.4 and B.2.4, the quantities in square brackets correspond to the components of the deformation tensor $D_{jk} = (1/2)(\partial_j u_k + \partial_k u_j)$.

B.1 Cylindrical coordinates (s, ϕ, z)

B.1.1 Conversion to cartesian coordinates

$$x = s\cos\phi\ , \qquad y = s\sin\phi\ , \qquad s = \sqrt{x^2+y^2}\ , \qquad \phi = \mathrm{atan}(y/x)\ , \qquad z = z\ .$$

$$\hat{\mathbf{e}}_x = \cos\phi\hat{\mathbf{e}}_s - \sin\phi\hat{\mathbf{e}}_\phi\ , \qquad \hat{\mathbf{e}}_y = \sin\phi\hat{\mathbf{e}}_s + \cos\phi\hat{\mathbf{e}}_\phi\ ,$$

$$\hat{\mathbf{e}}_s = \cos\phi\hat{\mathbf{e}}_x + \sin\phi\hat{\mathbf{e}}_y\ , \qquad \hat{\mathbf{e}}_\phi = -\sin\phi\hat{\mathbf{e}}_x + \cos\phi\hat{\mathbf{e}}_y\ , \qquad \hat{\mathbf{e}}_z = \hat{\mathbf{e}}_z\ .$$

B.1.2 Infinitesimals

$$\mathrm{d}\boldsymbol{\ell} = \mathrm{d}s\hat{\mathbf{e}}_s + s\mathrm{d}\phi\hat{\mathbf{e}}_\phi + \mathrm{d}z\hat{\mathbf{e}}_z$$

$$\mathrm{d}V = s\,\mathrm{d}s\,\mathrm{d}\phi\,\mathrm{d}z$$

B.1.3 Vector operators

$$\frac{D}{Dt} = \frac{\partial}{\partial t} + u_s\frac{\partial}{\partial s} + \frac{u_\phi}{s}\frac{\partial}{\partial\phi} + u_z\frac{\partial}{\partial z}$$

$$\nabla f = \frac{\partial f}{\partial s}\hat{\mathbf{e}}_s + \frac{1}{s}\frac{\partial f}{\partial\phi}\hat{\mathbf{e}}_\phi + \frac{\partial f}{\partial z}\hat{\mathbf{e}}_z$$

$$(\mathbf{u}\cdot\nabla)\mathbf{A} = \left(\mathbf{u}\cdot\nabla A_s - \frac{u_\phi A_\phi}{s}\right)\hat{\mathbf{e}}_s + \left(\mathbf{u}\cdot\nabla A_\phi + \frac{u_\phi A_s}{s}\right)\hat{\mathbf{e}}_\phi + (\mathbf{u}\cdot\nabla A_z)\,\hat{\mathbf{e}}_z$$

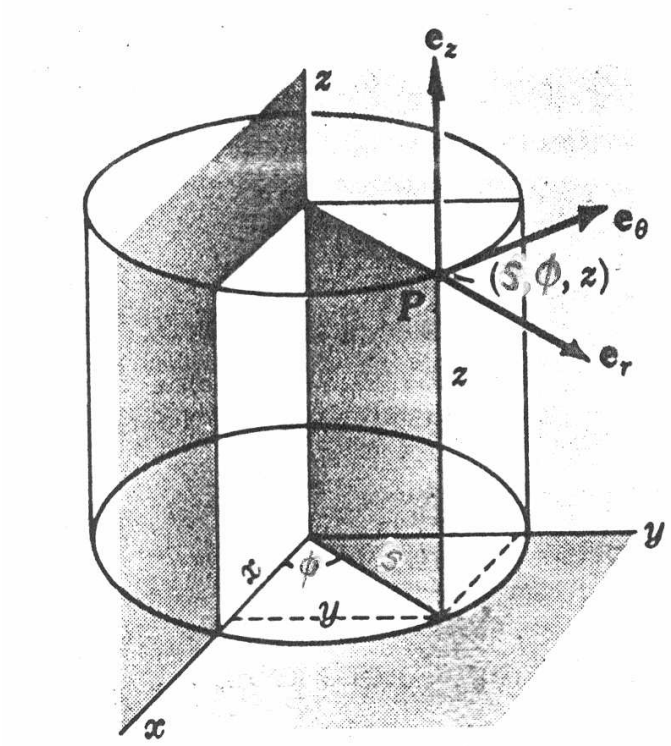

Figure B.1: Geometric definition of cylindrical coordinates. The transformation to Cartesian coordinates is given by $(x, y, z) = (s\cos\phi, s\sin\phi, z)$.

$$\nabla \cdot \mathbf{A} = \frac{1}{s}\frac{\partial}{\partial s}(sA_s) + \frac{1}{s}\frac{\partial A_\phi}{\partial \phi} + \frac{\partial A_z}{\partial z}$$

$$\nabla \times \mathbf{A} = \left(\frac{1}{s}\frac{\partial A_z}{\partial \phi} - \frac{\partial A_\phi}{\partial z}\right)\hat{\mathbf{e}}_s + \left(\frac{\partial A_s}{\partial z} - \frac{\partial A_z}{\partial s}\right)\hat{\mathbf{e}}_\phi + \frac{1}{s}\left(\frac{\partial (sA_\phi)}{\partial s} - \frac{\partial A_s}{\partial \phi}\right)\hat{\mathbf{e}}_z$$

$$\nabla^2 = \frac{1}{s}\frac{\partial}{\partial s}\left(s\frac{\partial}{\partial s}\right) + \frac{1}{s^2}\frac{\partial^2}{\partial \phi^2} + \frac{\partial^2}{\partial z^2}$$

$$\nabla^2\mathbf{A} = \left(\nabla^2 A_s - \frac{A_s}{s^2} - \frac{2}{s^2}\frac{\partial A_\phi}{\partial \phi}\right)\hat{\mathbf{e}}_s + \left(\nabla^2 A_\phi - \frac{A_\phi}{s^2} + \frac{2}{s^2}\frac{\partial A_s}{\partial \phi}\right)\hat{\mathbf{e}}_\phi + \left(\nabla^2 A_z\right)\hat{\mathbf{e}}_z$$

B.1.4 Components of the viscous stress tensor

$$\tau_{ss} = 2\mu\left[\frac{\partial u_s}{\partial s}\right] + (\mu_\vartheta - \frac{2}{3}\mu)\nabla \cdot \mathbf{u}$$

$$\tau_{\phi\phi} = 2\mu\left[\frac{1}{s}\frac{\partial u_\phi}{\partial \phi} + \frac{u_s}{s}\right] + (\mu_\vartheta - \frac{2}{3}\mu)\nabla \cdot \mathbf{u}$$

$$\tau_{zz} = 2\mu \left[\frac{\partial u_z}{\partial z} \right] + (\mu_\vartheta - \frac{2}{3}\mu) \nabla \cdot \mathbf{u}$$

$$\tau_{s\phi} = \tau_{\phi s} = 2\mu \left[\frac{1}{2} \left(\frac{1}{s} \frac{\partial u_s}{\partial \phi} + s \frac{\partial}{\partial s} \frac{u_\phi}{s} \right) \right]$$

$$\tau_{\phi z} = \tau_{z\phi} = 2\mu \left[\frac{1}{2} \left(\frac{\partial u_\phi}{\partial z} + \frac{1}{s} \frac{\partial u_z}{\partial \phi} \right) \right]$$

$$\tau_{zs} = \tau_{sz} = 2\mu \left[\frac{1}{2} \left(\frac{\partial u_z}{\partial s} + \frac{\partial u_s}{\partial z} \right) \right]$$

B.1.5 Equations of motion

$$\rho \left(\frac{\mathrm{D}u_s}{\mathrm{D}t} - \frac{u_\phi^2}{s} \right) = -\rho \frac{\partial \Phi}{\partial s} - \frac{\partial p}{\partial s} + \frac{1}{s} \frac{\partial}{\partial s} (s\tau_{ss}) + \frac{1}{s} \frac{\partial \tau_{s\phi}}{\partial \phi} + \frac{\partial \tau_{sz}}{\partial z} - \frac{\tau_{\phi\phi}}{s}$$

$$\rho \left(\frac{\mathrm{D}u_\phi}{\mathrm{D}t} - \frac{u_\phi u_s}{s} \right) = -\frac{\rho}{s} \frac{\partial \Phi}{\partial \phi} - \frac{1}{s} \frac{\partial p}{\partial \phi} + \frac{1}{s} \frac{\partial}{\partial s} (s\tau_{\phi s}) + \frac{1}{s} \frac{\partial \tau_{\phi\phi}}{\partial \phi} + \frac{\partial \tau_{\phi z}}{\partial z} + \frac{\tau_{s\phi}}{s}$$

$$\rho \frac{\mathrm{D}u_z}{\mathrm{D}t} = -\rho \frac{\partial \Phi}{\partial z} - \frac{\partial p}{\partial z} + \frac{1}{s} \frac{\partial}{\partial s} (s\tau_{zs}) + \frac{1}{s} \frac{\partial \tau_{z\phi}}{\partial \phi} + \frac{\partial \tau_{zz}}{\partial z}$$

B.1.6 The energy equation

$$\rho T \frac{\mathrm{D}s}{\mathrm{D}t} = \Phi_u + \frac{1}{s} \frac{\partial}{\partial s} \left[\chi s \frac{\partial T}{\partial s} \right] + \frac{1}{s^2} \frac{\partial}{\partial \phi} \left[\chi \frac{\partial T}{\partial \phi} \right] + \frac{\partial}{\partial z} \left[\chi \frac{\partial T}{\partial z} \right]$$

$$\Phi_u = 2\mu (D_{ss}^2 + D_{\phi\phi}^2 + D_{zz}^2 + 2D_{s\phi}^2 + 2D_{\phi z}^2 + 2D_{zs}^2) + (\mu_\vartheta - \frac{2}{3}\mu)(\nabla \cdot \mathbf{u})^2$$

B.2 Spherical coordinates (r, θ, ϕ)

B.2.1 Conversion to cartesian coordinates

$$x = r \sin\theta \cos\phi \ , \qquad y = r \sin\theta \sin\phi \ , \qquad z = r \cos\theta \ .$$

$$r = \sqrt{x^2 + y^2 + z^2} \ , \qquad \theta = \mathrm{atan}(\sqrt{x^2 + y^2}/z) \ , \qquad \phi = \mathrm{atan}(y/x) \ .$$

$$\hat{\mathbf{e}}_x = \sin\theta \cos\phi \hat{\mathbf{e}}_r + \cos\theta \cos\phi \hat{\mathbf{e}}_\theta - \sin\phi \hat{\mathbf{e}}_\phi \ ,$$

$$\hat{\mathbf{e}}_y = \sin\theta \sin\phi \hat{\mathbf{e}}_r + \cos\theta \sin\phi \hat{\mathbf{e}}_\theta + \cos\phi \hat{\mathbf{e}}_\phi \ ,$$

$$\hat{\mathbf{e}}_z = \cos\theta \hat{\mathbf{e}}_r - \sin\theta \hat{\mathbf{e}}_\theta \ .$$

$$\hat{\mathbf{e}}_r = \sin\theta \cos\phi \hat{\mathbf{e}}_x + \sin\theta \sin\phi \hat{\mathbf{e}}_y + \cos\theta \hat{\mathbf{e}}_z \ ,$$

$$\hat{\mathbf{e}}_\theta = \cos\theta \cos\phi \hat{\mathbf{e}}_x + \cos\theta \sin\phi \hat{\mathbf{e}}_y - \sin\theta \hat{\mathbf{e}}_z \ ,$$

$$\hat{\mathbf{e}}_\phi = -\sin\phi \hat{\mathbf{e}}_x + \cos\phi \hat{\mathbf{e}}_y \ .$$

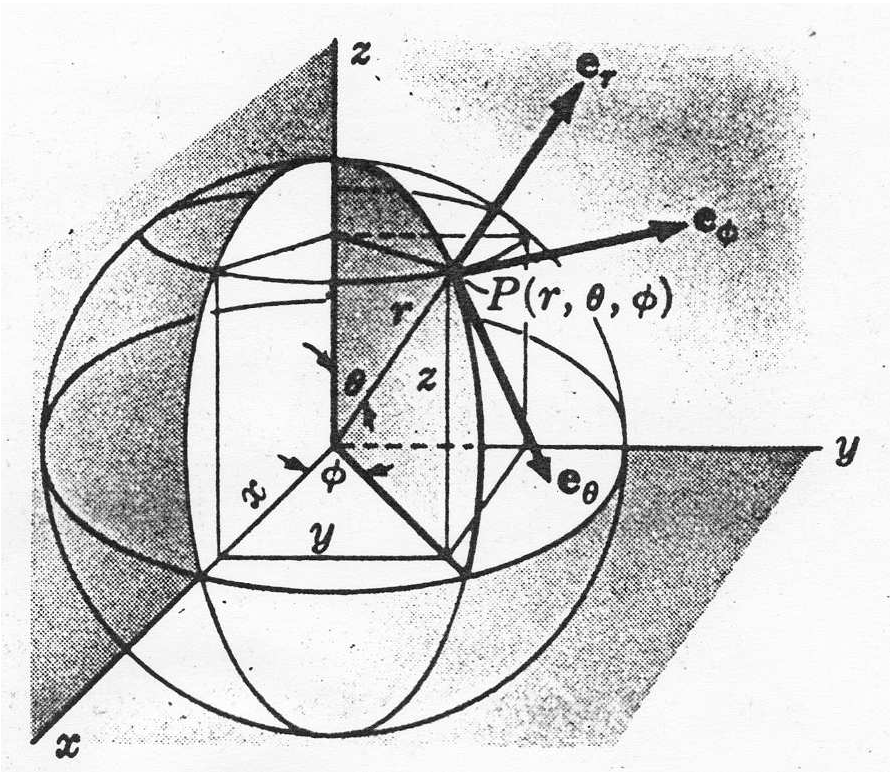

Figure B.2: Geometric definition of polar spherical copoprdinates. Transformation to Cartesian coordinates is given by $(x, y, z) = (r\sin\theta\cos\phi, r\sin\theta\sin\phi, r\cos\theta)$. Note that in so-called geographical coordinates, *longitude* $\equiv \phi$, but *latitude* $\equiv \pi/2 - \theta$.

B.2.2 Infinitesimals

$$\mathrm{d}\boldsymbol{\ell} = \mathrm{d}r\hat{\mathbf{e}}_r + r\mathrm{d}\theta\hat{\mathbf{e}}_\theta + r\sin\theta\mathrm{d}\phi\hat{\mathbf{e}}_\phi$$

$$\mathrm{d}V = r^2\sin\theta\,\mathrm{d}r\,\mathrm{d}\theta\,\mathrm{d}\phi$$

B.2.3 Operators

$$\frac{D}{Dt} = \frac{\partial}{\partial t} + u_r\frac{\partial}{\partial r} + \frac{u_\theta}{r}\frac{\partial}{\partial\theta} + \frac{u_\phi}{r\sin\theta}\frac{\partial}{\partial\phi}$$

$$\nabla f = \frac{\partial f}{\partial r}\hat{\mathbf{e}}_r + \frac{1}{r}\frac{\partial f}{\partial\theta}\hat{\mathbf{e}}_\theta + \frac{1}{r\sin\theta}\frac{\partial f}{\partial\phi}\hat{\mathbf{e}}_\phi$$

$$(\mathbf{u}\cdot\nabla)\mathbf{A} = \left(\mathbf{u}\cdot\nabla A_r - \frac{u_\theta A_\theta}{r} - \frac{u_\phi A_\phi}{r}\right)\hat{\mathbf{e}}_r$$

$$+\left(\mathbf{u}\cdot\nabla A_\theta - \frac{u_\phi A_\phi}{r}\cot\theta + \frac{u_\theta A_r}{r}\right)\hat{\mathbf{e}}_\theta + \left(\mathbf{u}\cdot\nabla A_\phi + \frac{u_\phi A_r}{r} + \frac{u_\phi A_\theta}{r}\cot\theta\right)\hat{\mathbf{e}}_\phi$$

$$\nabla \cdot \mathbf{A} = \frac{1}{r^2}\frac{\partial}{\partial r}(r^2 A_r) + \frac{1}{r\sin\theta}\frac{\partial(A_\theta \sin\theta)}{\partial\theta} + \frac{1}{r\sin\theta}\frac{\partial A_\phi}{\partial\phi}$$

$$\nabla \times \mathbf{A} = \frac{1}{r\sin\theta}\left(\frac{\partial(A_\phi \sin\theta)}{\partial\theta} - \frac{\partial A_\theta}{\partial\phi}\right)\hat{\mathbf{e}}_r$$

$$+\frac{1}{r\sin\theta}\left(\frac{\partial A_r}{\partial\phi} - \frac{\partial(A_\phi r\sin\theta)}{\partial r}\right)\hat{\mathbf{e}}_\theta + \frac{1}{r}\left(\frac{\partial(rA_\theta)}{\partial r} - \frac{\partial A_r}{\partial\theta}\right)\hat{\mathbf{e}}_\phi$$

$$\nabla^2 = \frac{1}{r^2}\frac{\partial}{\partial r}\left(r^2\frac{\partial}{\partial r}\right) + \frac{1}{r^2\sin\theta}\frac{\partial}{\partial\theta}\left(\sin\theta\frac{\partial}{\partial\theta}\right) + \frac{1}{r^2\sin^2\theta}\frac{\partial^2}{\partial\phi^2}$$

$$\nabla^2\mathbf{A} = \left(\nabla^2 A_r - \frac{2A_r}{r^2} - \frac{2}{r^2\sin\theta}\frac{\partial A_\theta\sin\theta}{\partial\theta} - \frac{2}{r^2\sin\theta}\frac{\partial A_\phi}{\partial\phi}\right)\hat{\mathbf{e}}_r$$

$$+\left(\nabla^2 A_\theta + \frac{2}{r^2}\frac{\partial A_r}{\partial\theta} - \frac{A_\theta}{r^2\sin^2\theta} - \frac{2\cos\theta}{r^2\sin^2\theta}\frac{\partial A_\phi}{\partial\phi}\right)\hat{\mathbf{e}}_\theta$$

$$+\left(\nabla^2 A_\phi + \frac{2}{r^2\sin\theta}\frac{\partial A_r}{\partial\phi} + \frac{2\cos\theta}{r^2\sin^2\theta}\frac{\partial A_\theta}{\partial\phi} - \frac{A_\phi}{r^2\sin^2\theta}\right)\hat{\mathbf{e}}_\phi$$

B.2.4 Components of the viscous stress tensor

$$\tau_{rr} = 2\mu\left[\frac{\partial u_r}{\partial r}\right] + (\mu_\vartheta - \frac{2}{3}\mu)\nabla\cdot\mathbf{u}$$

$$\tau_{\theta\theta} = 2\mu\left[\frac{1}{r}\frac{\partial u_\theta}{\partial\theta} + \frac{u_r}{r}\right] + (\mu_\vartheta - \frac{2}{3}\mu)\nabla\cdot\mathbf{u}$$

$$\tau_{\phi\phi} = 2\mu\left[\frac{1}{r\sin\theta}\frac{\partial u_\phi}{\partial\phi} + \frac{u_r}{r} + \frac{u_\theta\cot\theta}{r}\right] + (\mu_\vartheta - \frac{2}{3}\mu)\nabla\cdot\mathbf{u}$$

$$\tau_{r\theta} = \tau_{\theta r} = 2\mu\left[\frac{1}{2}\left(\frac{1}{r}\frac{\partial u_r}{\partial\theta} + r\frac{\partial}{\partial r}\frac{u_\theta}{r}\right)\right]$$

$$\tau_{\theta\phi} = \tau_{\phi\theta} = 2\mu\left[\frac{1}{2}\left(\frac{1}{r\sin\theta}\frac{\partial u_\theta}{\partial\phi} + \frac{\sin\theta}{r}\frac{\partial}{\partial\theta}\frac{u_\phi}{\sin\theta}\right)\right]$$

$$\tau_{\phi r} = \tau_{r\phi} = 2\mu\left[\frac{1}{2}\left(r\frac{\partial}{\partial r}\frac{u_\phi}{r} + \frac{1}{r\sin\theta}\frac{\partial u_r}{\partial\phi}\right)\right]$$

B.2.5 Equations of motion

$$\rho\left(\frac{\mathrm{D}u_r}{\mathrm{D}t}-\frac{u_\theta^2+u_\phi^2}{r}\right)=-\rho\frac{\partial\Phi}{\partial r}-\frac{\partial p}{\partial r}$$

$$+\frac{1}{r\sin\theta}\left[\frac{\sin\theta}{r}\frac{\partial}{\partial r}(r^2\tau_{rr})+\frac{\partial}{\partial\theta}(\tau_{r\theta}\sin\theta)+\frac{\partial\tau_{r\phi}}{\partial\phi}\right]-\frac{\tau_{\theta\theta}+\tau_{\phi\phi}}{r}$$

$$\rho\left(\frac{\mathrm{D}u_\theta}{\mathrm{D}t}+\frac{u_ru_\theta}{r}-\frac{u_\phi^2\cot\theta}{r}\right)=-\frac{\rho}{r}\frac{\partial\Phi}{\partial\theta}-\frac{1}{r}\frac{\partial p}{\partial\theta}$$

$$+\frac{1}{r\sin\theta}\left[\frac{\sin\theta}{r}\frac{\partial}{\partial r}(r^2\tau_{\theta r})+\frac{\partial}{\partial\theta}(\tau_{\theta\theta}\sin\theta)+\frac{\partial\tau_{\theta\phi}}{\partial\phi}\right]+\frac{\tau_{r\theta}}{r}-\frac{\tau_{\phi\phi}\cot\theta}{r}$$

$$\rho\left(\frac{\mathrm{D}u_\phi}{\mathrm{D}t}+\frac{u_ru_\phi}{r}+\frac{u_\theta u_\phi\cot\theta}{r}\right)=-\frac{\rho}{r\sin\theta}\frac{\partial\Phi}{\partial\phi}-\frac{1}{r\sin\theta}\frac{\partial p}{\partial\phi}$$

$$+\frac{1}{r\sin\theta}\left[\frac{\sin\theta}{r}\frac{\partial}{\partial r}(r^2\tau_{\phi r})+\frac{\partial}{\partial\theta}(\tau_{\phi\theta}\sin\theta)+\frac{\partial\tau_{\phi\phi}}{\partial\phi}\right]+\frac{\tau_{r\phi}}{r}+\frac{\tau_{\theta\phi}\cot\theta}{r}$$

B.2.6 The energy equation

$$\rho T\frac{\mathrm{D}s}{\mathrm{D}t}=\Phi_u+\frac{1}{r^2}\frac{\partial}{\partial r}\left[\chi r^2\frac{\partial T}{\partial r}\right]+\frac{1}{r^2\sin\theta}\frac{\partial}{\partial\theta}\left[\chi\sin\theta\frac{\partial T}{\partial\theta}\right]+\frac{1}{r^2\sin^2\theta}\frac{\partial}{\partial\phi}\left[\chi\frac{\partial T}{\partial\phi}\right]$$

$$\Phi_u=2\mu(D_{rr}^2+D_{\theta\theta}^2+D_{\phi\phi}^2+2D_{r\theta}^2+2D_{\theta\phi}^2+2D_{\phi r}^2)+(\mu_\vartheta-\frac{2}{3}\mu)(\nabla\cdot\mathbf{u})^2$$

Bibliography:

For more on all this stuff, see

P.M. Morse et H. Feshbach, *Methods of Theoretical Physics*, McGraw-Hill (1953): Chap. 1
G.K. Batchelor, *An Introduction to Fluid Dynamics*, Cambridge University Press (1967): Appendice 2
Arfken, G., *Mathematical Methods for Physicists*, 2e éd., Academic Press (1970): chap. 2.

or, for a concise introduction, Appendix A.2 of the Goedbloed & Poedts tome cited in the bibliography to chapter 1.

Appendix C

Physical and astronomical constants

C.1 Physical constants

Table C.1

Physical Quantity	Symbol	Value	Units (SI)
Charge of electron	e	1.602×10^{-19}	C
Mass of electron	m_e	9.109×10^{-31}	kg
Mass of proton	m_p	1.673×10^{-27}	kg
Permittivity of vacuum	ε_0	8.854×10^{-12}	
Permeability of vacuum	μ_0	$4\pi \times 10^{-7}$	
Speed of light	c	2.998×10^{8}	m s^{-1}
Planck constant	h	6.626×10^{-34}	J s
Boltzmann constant	h	1.381×10^{-23}	J K^{-1}
Stefan-Boltzmann constant	σ	5.670×10^{-YY}	J $K^{-4}m^{-2}s^{-1}$
Gravitational constant	G	6.671×10^{-11}	$m^3kg^{-1}s^{-2}$

C.2 Astronomical constants

Table C.2

Astronomical Quantity	Symbol	Value	Units (SI)
Earth mass	$M_\oplus$	5.977×10^{24}	kg
Earth radius	$R_\oplus$	6.378×10^{6}	m
Astronomical Unit	AU	1.496×10^{11}	m
Solar mass	$M_\odot$	1.989×10^{30}	kg
Solar radius	$R_\odot$	6.960×10^{8}	m
Solar luminosity	$L_\odot$	3.83×10^{26}	J s^{-1}
Parsec	pc	3.086×10^{16}	m
Light-year	ly	9.461×10^{15}	m

Appendix D

Maxwell's equations and physical units

Electromagnetism is, unfortunately, a subfield of physics where the choice of units does not only influence the numerical values assigned to measurements, but also the mathematical form of the fundamental laws, i.e., Maxwell's equations.

D.1 Maxwell's equations

The whole mess in converting SI units to the astrophysically ubiquitous CGS units all harks back to the definition for the unit of charge, as embodied in Coulomb's Law. Under the SI system we write the electrostatic force between two charges q_1 and q_2 located at positions $\mathbf{x}_1$ and $\mathbf{x}_2$ as

$$\mathbf{F} = \frac{1}{4\pi\varepsilon_0}\frac{q_1 q_2}{r^2}\hat{\mathbf{r}} \ , \qquad [\text{SI}] \ , \tag{D.1}$$

with electrical charge measured in coulomb, and with $\mathbf{r} \equiv \mathbf{x}_1 - \mathbf{x}_2$ for notational brevity. whereas under the CGS system the constant $1/4\pi\varepsilon_0$ is absorbed into the definition of the unit of charge:

$$\mathbf{F} = \frac{q_1 q_2}{(\mathbf{x}_1 - \mathbf{x}_2)^2} \qquad [\text{CGS}] \ , \tag{D.2}$$

with electrical charge now measured in "electrostatic units", abbreviated "esu" and sometimes also called "statcoulomb". It electrostatics it is relatively easy to switch from CGS to SI with the simple substitution $\varepsilon_0 \to 1/(4\pi)$. With electrical currents now measured in esu s^{-1} in the CGS system, and remembering that $c^2 = (\varepsilon_0\mu_0)^{-1}$, the $\mu_0/4\pi$ prefactor in the Biot-Savart Law now becomes $1/c$:

$$\mathbf{B} = \frac{1}{c}\int \frac{\mathrm{d}\boldsymbol{\ell} \times \mathbf{r}}{r^2} \ , \qquad [\text{CGS}] \ , \qquad [\text{Biot} - \text{Savart}] \tag{D.3}$$

If you then now go through the process of re-constructing Maxwell's equations under these two new forms for the fundamental relations (electric and magnetic forces), you eventually get to

$$\nabla \cdot \mathbf{E} = 4\pi\rho_e \ , \qquad [\text{Gauss}' \text{ Law}] \tag{D.4}$$

$$\nabla \cdot \mathbf{B} = 0 \ , \qquad [\text{Anonymous}] \tag{D.5}$$

$$\nabla \times \mathbf{E} = -\frac{1}{c}\frac{\partial \mathbf{B}}{\partial t} \ , \qquad [\text{Faraday}'\text{s Law}] \tag{D.6}$$

$$\nabla \times \mathbf{B} = \frac{4\pi}{c}\mathbf{J} + \frac{1}{c}\frac{\partial \mathbf{E}}{\partial t} \ , \qquad \text{[Ampere/Maxwell's Law)]} \tag{D.7}$$

In some sense, the CGS system is perhaps more "natural", as it omits the introduction of new, apparently fundamental physical constants ε_0 and μ_0, to simply stick with the speed of light c, the only price to pay being an extraneous factor 4π in Gauss' Law. The Lorentz force and Poynting vector become, in CGS units:

$$\mathbf{F} = q(\mathbf{E} + \frac{1}{c}\mathbf{u} \times \mathbf{B}) \qquad \text{[Lorentz Force]} \tag{D.8}$$

$$\mathbf{S} = \frac{c}{4\pi}(\mathbf{E} \times \mathbf{B}) \qquad \text{[Poynting]} \tag{D.9}$$

and the electrostatoic and magnetic energies:

$$\mathcal{E}_e = \frac{1}{8\pi}\int \mathrm{E}^2 \mathrm{d}V \ , \tag{D.10}$$

$$\mathcal{E}_B = \frac{1}{8\pi}\int \mathrm{B}^2 \mathrm{d}V \ . \tag{D.11}$$

D.2 Conversion of units

The Table that follows gives you the conversion factor (f) required to go **from** SI **to** cgs units, i.e., SI Unit = $f \times$ cgs units. Any "3" appearing in a given value for f is a notational shortcut for 2.99792458.

Table D.1
Conversion between SI ands CGS units

Quantity	SI name	SI symbol	conversion factor f	CGS name	CGS symbol
Length	meter	m	10^2	centimeter	cm
Mass	kilogram	kg	10^3	gram	g
Force	newton	N	10^5	dyne	dyne
Energy	joule	J	10^7	erg	erg
Charge	coulomb	C	3×10^9	electrostatic units	esu
Current	ampere	A	3×10^9	statampere	esu s^{-1}
Potential	volt	V	$1/300$	statvolt	statvolt
Electric field	—	V m^{-1}	$(1/3)\times 10^{-4}$	—	statvolt cm^{-1}
Magnetic field	tesla	T	10^4	gauss	G
Magnetic flux	weber	Wb	10^8	maxwell	Mx

For a somewhat humourous close to this rather dry Appendix, here are five different ways, actually to be found in various textbooks or research monographs, to express teslas in terms of other fundamental SI units:

$$1\,\mathrm{T} = 1\,\frac{\mathrm{V\,s}}{\mathrm{m}^2} = 1\,\frac{\mathrm{N}}{\mathrm{A\,m}} = 1\,\frac{\mathrm{kg}}{\mathrm{A\,s}^2} = 1\,\frac{\mathrm{Wb}}{\mathrm{m}^2} = 1\,\frac{\mathrm{kg}}{\mathrm{C\,s}} \ . \tag{D.12}$$

Bibliography:

The content of this Appendix is taken primarily from Appendix C in

Griffith, D.J., *Introduction to Electrodynamics*, 3rd ed., Prentice Hall (1999).

Appendix E

The polytropic approximation

In practical terms, the polytropic approximation is used to replace the energy equation by a simple algebraic expression relating pressure and density (we do need as many equations as unknown functionals...).

The starting point is the assumption that at every point in the gas there exist a linear relationship between fractional variations between pressure and density:

$$\frac{\mathrm{d}p}{p} = \frac{\mathrm{d}\rho}{\rho} \ , \tag{E.1}$$

or, equivalently

$$\frac{\mathrm{d}\log p}{\mathrm{d}\log \rho} = \alpha \ , \tag{E.2}$$

which integrates directly to

$$\frac{p}{p_0} = \left(\frac{\rho}{\rho_0}\right)^{\alpha} \ , \tag{E.3}$$

which does achieve or stated goal, which is to express p as some function of ρ, but what does it mean physically? From the equation of state for a perfect gas, one can easily obtain

$$\frac{\mathrm{d}\log p}{\mathrm{d}\log \rho} = \left(1 - \frac{c_p - c_v}{p}\frac{\mathrm{d}T}{\mathrm{d}V}\right) \ , \tag{E.4}$$

where the second equality results from use Carnot's Law $R = c_p - c_v$, remembering that the specific volume V is equivalent to ρ^{-1}. Comparing the RHS of this expression to that of eq. (E.2), it is clear that α is related ot the thermodynamical properties of the substance under consideration.

Now recall that the second law of thermodynamics states that any local heat input $\mathrm{d}Q$ goes either into increasing the specific energy $\mathrm{d}U$ of a gas, or into work against the ambient pressure ($p\mathrm{d}V$):

$$\mathrm{d}Q = \mathrm{d}U + p\mathrm{d}V \ . \tag{E.5}$$

From a macroscopic point of view, the specific heat C is defined via the relation

$$\mathrm{d}Q = C\mathrm{d}T \ , \tag{E.6}$$

and gives a measure of the amount of heating/cooling required to achieve a temperature change of magnitude $\mathrm{d}T$. We also have

$$\mathrm{d}U = c_v\mathrm{d}T \ , \tag{E.7}$$

Using these two expression, the Second Law can be manipulated into

$$\frac{\mathrm{d}V}{\mathrm{d}T} = \frac{C - c_v}{p} \ , \tag{E.8}$$

so that eq. (E.4) becomes

$$\frac{\mathrm{d}\log p}{\mathrm{d}\log \rho} = \frac{C - c_p}{C - c_v} \ , \tag{E.9}$$

implying

$$\alpha = \frac{C - c_p}{C - c_v} \ . \tag{E.10}$$

You may recall from statistical thermodynamics that c_p and c_v are related to the number of degrees of freedom available in interacting with other particles. The quantity C, on the other hand, is a resolutely macroscopic beast measuring the thermodynamic behavior of the system as a whole. In adiabatic systems, no heat flow in or out of the system is allowed, i.e., $\mathrm{d}Q = 0$. The only way temperature actually change without violating eq. (E.6) is to have $C = 0$. The opposite bound is the isothermal limit, in which even a $\mathrm{d}Q$ tending to infinity cannot change the temperature. This can only be accomodated within eq. (E.6) if we can let $C \to \infty$. Equation (E.10) can be used to translate these bounds on C into bounds on α:

$$\text{[isothermal]} \qquad 1 \leq \alpha \leq c_p/c_v \qquad \text{[adiabatic]} \tag{E.11}$$

The ratio of specific heats c_p/c_v is equal to 5/3 for a non-relativistic perfect gas; the allowed range of α is quite restricted!

Except for assuming a perfect gas equation of state, we have made no approximation until now, all we have done is manipulate thermodynamic relationships. Our starting point, eq. (E.1) holds formally true for any perfect gas with constant mean molecular weight. The actual value of α is a function of local heat input and drain, which may well be a function of space and time (and constrained by eq. (E.11)). So we haven't accomplished anything really, we just replaced one unknown functional, $p(\mathbf{x}, t)$, by another, $\alpha(\mathbf{x}, t)$.

The **polytropic approximation** consists in assuming not only that α is a constant in space and time, but moreover that the value of this constant can be set *a priori* within the bounds set by eq. (E.11). Evidently, this amounts to assuming the presence of some very specific profile of heat source/drain in the system. This is normally something we would get out of the energy equation, so in that sense the polytropic approximation is indeed replacing the energy equation.

It often proves convenient to rewrite eq. (E.3) in terms of a polytropic sound speed c_s, defined via

$$\alpha p = c_s^2 \rho \ , \tag{E.12}$$

or, alternately,

$$c_s^2 = c_{s0}^2 \left(\frac{\rho}{\rho_0}\right)^{\alpha - 1} \ , \tag{E.13}$$

with $c_{s0}^2 = \alpha p/\rho_0$. In the isothermal limit we have

$$p = a^2 \rho \ , \tag{E.14}$$

where $a = \sqrt{kT/\mu m}$ is the isothermal sound speed, and a constant for fixed mean molecular weight. It is left as an exercise to show that under the polytropic approximation the pressure gradient term in the momentum equation can be written as

$$\frac{1}{\rho}\nabla p = c_s^2 \nabla \left(\frac{\rho}{\rho_0}\right) \ . \tag{E.15}$$

Note finally that in doing so, we haven't just eliminated the energy equation, but also absorbed the equation of state; the pressure is now to be computed directly from ρ, without explicit need for temperature.

Appendix F

Essential numerics

This chapter collects various numerical algorithms that may prove useful in working through some of the problems scattered throughout the class notes. The emphasis is on robust, easy-to-code algorithms, rather than on algorithms that are very efficient (in terms of operation count) of very accurate (in terms of theoretical convergence rates). In most instances the algorithms are stated with very little formal justification, but pointers are given to various numerical analysis books that should satisfy the eager beaver.

F.1 Derivatives: finite differences

The task is to evaluate derivatives of a continuous function $f(x)$ of one variable x. We begin by *discretizing* $f(x)$, by sampling it on a discrete (1-D) spatial mesh

$$x \to \{x_1, x_2, ..., x_N\}, \qquad [x_{j+1} > x_j] \ , \tag{F.1}$$

with constant mesh increment $h \equiv x_{j+1} - x_j$. Consider now the two following Taylor series expansions for $f(x_j \pm h)$ about $f(x_j)$:

$$f(x_j + h) \equiv f(x_{j+1}) = f(x_j) + h\frac{\mathrm{d}f}{\mathrm{d}x}\bigg|_{x_j} + \frac{h^2}{2!}\frac{\mathrm{d}^2 f}{\mathrm{d}x^2}\bigg|_{x_j} + \frac{h^3}{3!}\frac{\mathrm{d}^3 f}{\mathrm{d}x^3}\bigg|_{x_j} + ... \tag{F.2}$$

$$f(x_j - h) \equiv f(x_{j-1}) = f(x_j) - h\frac{\mathrm{d}f}{\mathrm{d}x}\bigg|_{x_j} + \frac{h^2}{2!}\frac{\mathrm{d}^2 f}{\mathrm{d}x^2}\bigg|_{x_j} - \frac{h^3}{3!}\frac{\mathrm{d}^3 f}{\mathrm{d}x^3}\bigg|_{x_j} + ... \tag{F.3}$$

These two expressions can be rerranged in the form

$$\frac{\mathrm{d}f}{\mathrm{d}x}\bigg|_{x_j} = \frac{f(x_{j+1}) - f(x_j)}{h} - \frac{h}{2!}\frac{\mathrm{d}^2 f}{\mathrm{d}x^2}\bigg|_{x_j} - \frac{h^2}{3!}\frac{\mathrm{d}^3 f}{\mathrm{d}x^3}\bigg|_{x_j} - ... \tag{F.4}$$

$$\frac{\mathrm{d}f}{\mathrm{d}x}\bigg|_{x_j} = \frac{f(x_j) - f(x_{j-1})}{h} + \frac{h}{2!}\frac{\mathrm{d}^2 f}{\mathrm{d}x^2}\bigg|_{x_j} - \frac{h^2}{3!}\frac{\mathrm{d}^3 f}{\mathrm{d}x^3}\bigg|_{x_j} + ... \tag{F.5}$$

Mathematical approximations of derivatives, known as **finite differences**, are obtained by algebraic manipulations of eqs. (F.4)—(F.5); for example, adding the two, neglecting all terms proportional to powers in h of two and higher, and solving for $\partial f/\partial x$ readily yields

$$\frac{\mathrm{d}f}{\mathrm{d}x}\bigg|_{x_j} = \frac{f_{j+1} - f_{j-1}}{2h} + O(h^2) \ , \tag{F.6}$$

while subtracting (F.5) from (F.4) and solving for $\mathrm{d}^2 f/\mathrm{d}x^2$ leads to

$$\left.\frac{\mathrm{d}^2 f}{\mathrm{d}x^2}\right|_{x_j} = \frac{f_{j+1} - 2f_j + f_{j-1}}{h^2} + O(h^2) \ , \tag{F.7}$$

where, in both cases, $f_j \equiv f(x_j)$ for brevity of notation. Likewise, one can show that

$$\left.\frac{\mathrm{d}^3 f}{\mathrm{d}x^3}\right|_{x_j} = \frac{f_{j+2} - 2f_{j+1} + 2f_{j-1} - f_{j-2}}{2h^3} + O(h^2) \ , \tag{F.8}$$

$$\left.\frac{\mathrm{d}^4 f}{\mathrm{d}x^4}\right|_{x_j} = \frac{f_{j+2} - 4f_{j+1} + 6fj - 4f_{j-1} + f_{j-2}}{h^4} + O(h^2) \ . \tag{F.9}$$

The term $O(h^2)$ on the RHS of eqs. (F.6)–(F.7) means that the discretization error associated with the finite difference expression decreases as h^2 in the limit $h \to 0$, as per our neglect of all terms proportional to $h^2, h^3, ...$ in eqs. (F.4)–(F.5). The finite difference formulae given above are **centered** on node x_j. They evidently cannot be used at the first and last node of the mesh. However, manipulations of eqs. (F.4)–(F.4) can also yield **forward difference** formulae, e.g.:

$$\left.\frac{\mathrm{d}f}{\mathrm{d}x}\right|_{x_j} = \frac{f_{j+1} - f_j}{h} + O(h) \ , \tag{F.10}$$

$$\left.\frac{\mathrm{d}f}{\mathrm{d}x}\right|_{x_j} = \frac{-f_{j+2} + 4f_{j+1} - 3f_j}{2h} + O(h^2) \ , \tag{F.11}$$

$$\left.\frac{\mathrm{d}^2 f}{\mathrm{d}x^2}\right|_{x_j} = \frac{f_{j+2} - 2f_{j+1} + f_j}{h^2} + O(h) \ , \tag{F.12}$$

$$\left.\frac{\mathrm{d}^2 f}{\mathrm{d}x^2}\right|_{x_j} = \frac{-f_{j+3} + 4f_{j+2} - 5f_{j+1} + 2f_j}{h^2} + O(h^2) \ , \tag{F.13}$$

$$\left.\frac{\mathrm{d}^3 f}{\mathrm{d}x^3}\right|_{x_j} = \frac{f_{j+3} - 3f_{j+2} + 3f_{j+1} - f_j}{h^3} + O(h) \ , \tag{F.14}$$

$$\left.\frac{\mathrm{d}^4 f}{\mathrm{d}x^4}\right|_{x_j} = \frac{f_{j+4} - 4f_{j+3} + 6f_{j+2} - 4f_{j+1} + f_j}{h^4} + O(h) \ , \tag{F.15}$$

and similar **backward difference** formulae, which can be used at the endpoints of the mesh. Further manipulations of eqs. (F.4)–(F.5) and recurrent use of the $O(h)$ and $O(h^2)$ difference formulae leads to higher order expressions

$$\left.\frac{\partial f}{\partial x}\right|_{x_j} = \frac{-f_{j+1} + 8f_{j+1} - 8f_{j-1} + f_{j-1}}{12h} + O(h^4) \ , \tag{F.16}$$

$$\left.\frac{\mathrm{d}^2 f}{\mathrm{d}x^2}\right|_{x_j} = \frac{-f_{j+2} + 16f_{j+1} - 30f_j + 16f_{j-1} - f_{j-2}}{12h^2} + O(h^4) \ . \tag{F.17}$$

These basic ideas carry over to the computation of partial derivatives in more than one spatial dimension. For the lower order formulae the corresponding expressions are direct extensions of the formulae listed above, with other variables held constant. For example, on a 2D mesh

$$\left.\frac{\partial f}{\partial x}\right|_{x_j,y_k} = \frac{f_{j+1,k} - f_{j-1,k}}{2h} + O(h^2) \ , \tag{F.18}$$

where $f_{j,k} \equiv f(x_j, y_k)$. For higher order formulae, however, diagonal nodes (e.g. $f_{j+1,k+1}$) enter the expressions. For more details see §2.1 of the Lapidus & Pinder book cited in the bibliography.

In setting up finite differences for a given problem it is usually a good idea to use formulae having similar order of truncation error. It must be kept in mind that the theoretical truncation errors ($O(h^2)$, etc) hold only in the limit of small mesh size; if your spatial mesh underresolves the function being discretized, eqs. (F.16)–(F.17) will give results most likely as bad as eqs. (F.6)–(F.7) would.

F.2 Integrals: Numerical integration

In a nutshell, numerical integration proceeds by approximating the integrand by polynomials that can be integrated analytically. This is in fact mathematically equivalent to exact integration of a truncated Taylors series expansion of the integrand.

The **trapezoidal rule** is a simple and robust method that allows the numerical integration of a function $f(x)$ sampled over its integration domain on a mesh with intervals that may or may not be equidistant:

$$(x_1, x_2, x_3, ..., x_N) \ , \qquad x_{k+1} - x_k \neq x_k - x_{k-1} \ . \tag{F.19}$$

The underlying idea is simply to approximate the variations of the function $f(x)$ from one mesh point to the next by a linear function, amounting to a piecewise=continuous linear interpolation of $f(x)$:

$$\int_{x_1}^{x_N} f(x)\,\mathrm{d}x \simeq \sum_{k=1}^{N-1} \frac{(f_{k+1} + f_k)}{2}(x_{k+1} - x_k) \ , \tag{F.20}$$

with $f_k \equiv f(x_k)$. The tighter the mesh, the better the discrete sum on the RHS of this expression approximates the integral on the LHS. For an equidistant mesh, the discretization error of the trapezoidal method is formally $O(h^2)$.

The idea of interpolating between mesh points is readily generalized to higher-order interpolation. For an equidistant mesh ($x_{k+1} - x_k = h\,\forall k$), the resulting integration formulae (known as Newton-Cotes formulae) are relatively simple: [N-1 even]:

$$\int_{x_1}^{x_N} f(x)\,\mathrm{d}x = \frac{h}{3}(f_1 + 4f_2 + 2f_3 + 4f_4 + 2f_5 + ... + 4f_{N-1} + f_N) + O(h^4) \ , \tag{F.21}$$

[N-1 multiple of 3]:

$$\int_{x_1}^{x_N} f(x)\,\mathrm{d}x = \frac{3h}{8}(f_1 + 3f_2 + 3f_3 + 2f_4 + 3f_5 + 3f_6 ... + 3f_{N-1} + f_N) + O(h^4) \ , \tag{F.22}$$

The first of these relations is known as the **Simpson 1/3 Rule**, while the second is the **Simpson 3/8 Rule**. The mathematical developments leading to these formulae are detailed in chapter 4 of the book by Gerald cited in the bibliography.

Romberg integration is a really neat trick that produces great improvement on the accuracy of numerical integration, with very little extra work. Consider for example the use

of the trapezoidal rule on an equidistant mesh $h = h_1$. For a sufficiently tight mesh, the computed integral ($I^{(1)}$, say) will differ from the exact solution (I^*) by a factor proportional to h_1^2 to leading order; one can therefore write something like:

$$I^{(1)} = I^* + Ch_1^2 \ , \tag{F.23}$$

where C is a constant that is a property of the chosen integration algorithm, but not of the mesh. Consider now a second numerical evaluation of the integral, obtained on a mesh $h_2 \neq h_1$. Once again we have

$$I^{(2)} = I^* + Ch_2^2 \ , \tag{F.24}$$

with the same C. Equations (F.23) and (F.24) define a system of two algebraic equations for the two unknowns I^* et C; this system is readily solved for I^*:

$$I^* = \frac{h_2^2 I^{(1)} - h_1^2 I^{(2)}}{h_1^2 - h_2^2} \ , \qquad O(h^2) \text{ methods} \ . \tag{F.25}$$

This is in fact a form of extrapolation, so that it is a good idea to use a second mesh that is significantly different from the first, e.g., $h_2 = h_1/2$. For the higher-order Simpson rules, you can easily show that Romberg extrapolation now takes the form:

$$I^* = \frac{h_2^4 I^{(1)} - h_1^4 I^{(2)}}{h_1^4 - h_2^4} \ . \qquad O(h^4) \text{ methods} \tag{F.26}$$

F.3 Root finding: The bisection method

Finding zeros of a one-dimensional function is another common numerical task, one if fact that you will have encountered under many guises in part II of these class notes.

Given a nonlinear function of one variable $f(x)$ and a range $[x_1, x_2]$ in which we seek a root x_r such that $f(x_r) = 0$, within within a (predetermined) absolute accuracy ε of the true root. Among the many numerical techniques that can achieve this, the **bisection method** is probably the most robust and easy to code:

while $\delta \geq \varepsilon$ **do**
    $x_m = (x_1 + x_2)/2$
    if $f(x_2) \times f(x_m) \geq 0$ **then**
        $x_2 = x_m$
    else
        $x_1 = x_m$
    endif
    $\delta = x_2 - x_1$
enddo

The underlying logic behind this algorithm is simplicity itself: if there is a root at x_r, then $f(x)$ must have different sign on either side of x_r. The nice thing about the bisection method is that it is *guaranteed* to find the root *if* a root indeed exists in the (user-specified) interval $[x_1, x_2]$. Its chief drawback is that it converges linearly to the root, which, if high accuracy is required, can take a lot of iterations.

Chapter 9 of the book *Numerical Recipes* (see bibliography) contains a very accessible introduction to the numerical solution of nonlinear rootfinding problems, including the bisection method. You will also learn therein about a neat trick called **bracketing** that ensures that the initial bisection interval $[x_1, x_2]$ always contains at least one root (if any root at all exists in $x \in [-\infty, \infty]$).

F.4 Maximization of functions: hill climbing

Maximization and minimization are two faces of the same coin, namely optimization, since maximizing $f(\mathbf{x})$ is the same as minimizing $-f(\mathbf{x})$, or $1/f(\mathbf{x})$, etc. Minimization/maximization is one of the most common numerical task encountered in the physical sciences, for example when you are trying to fit a parametric mathematical relation to data.

In your first calculus class you have learned how to do this: find the $\mathbf{x}$ such that $\mathrm{d}f/\mathrm{d}x = 0$ and $\partial^2 f/\partial x^2 < 0$ (for maximization). But in practice this is almost never a practical approach, because even for a one-dimension search space the resulting (usually nonlinear) rootfinding problem turns out very, very hard. The dimension N of the search space is usually the key issue, with optimization problems in high-dimensional search spaces being usually much, much harder than in low-dimensional search spaces. In one dimension, the technique known as **Golden search**, an analog of bisection, is the method of choice as far as simplicity and robustness go. However, things get a lot trickier as soon as we hit two-dimensional search spaces.

Strictly speaking, maximization means finding the point $\mathbf{x}^*$ in some N-dimensional space so that $f(\mathbf{x}^*) > f(\mathbf{x})\ \forall \mathbf{x} \neq \mathbf{x}^*$. Here $\mathbf{x}$ could be a set of N parameters defining a model you are trying to fit to some data. The vast majority of multi-dimensional optimization techniques to be found in textbooks on numerical analysis or scientific computing all proceed by **hill climbing**. As the name implies, this method just follows the slope of $f(\mathbf{x})$ all the way to the "summit". A generic hill climbing scheme for maximization would look as follows:

Initialization: $\mathbf{x} = \mathbf{x}_0$
for $k = 1, 2, ...$ **do**
    Construct: $\Delta\mathbf{x}$
    $\mathbf{x}^* = \mathbf{x} + \Delta\mathbf{x}$
    $\delta = |f(\mathbf{x}^*) - f(\mathbf{x})|$
    if $\delta \geq \varepsilon$ **then**
        $\mathbf{x} = \mathbf{x}^*$
    else
        Maximum found: $\mathbf{x}^*$
        exit
    endif
enddo

Note that three things are needed here: (1) a starting guess $\mathbf{x}_0$; (2) a "recipe" for constructing a displacement $\Delta\mathbf{x}$ in parameter space; and (3) a stopping criterion. Here the latter is defined in terms of the absolute accuracy ε in evaluating $f(\mathbf{x}^*)$, but other termination criteria involving instead $\mathbf{x}^* - \mathbf{x}$ are possible. Of course the crux here is to construct a $\Delta\mathbf{x}$ that will make $f(\mathbf{x}^*) > f(\mathbf{x}$. Indeed, all these fancy optimization schemes you will find in the literature mostly differ in how they go about constructing $\Delta\mathbf{x}$ (i.e., choosing a direction in parameter space and a step length), and whether and how, in doing so, they make use of information accumulated from previous steps. These days most programming languages come equipped with with a library of canned optimization routines that usually perform well... on problems that are not too hard.

Hill climbing methods can be divided in two broad classes, namely those making use of gradient information, and those who do not. Generally speaking, the former tend to be more efficient, the latter more robust. One particularly simple way to pick $\Delta\mathbf{x}$, known as stochastic hill climbing, is to pick a random direction in the N-dimensional search space, and take a step with a length extracted from some suitable statistical distribution (usually peaked at zero, but with a significant tail). Do this ten times (say), each time evaluating $f(\mathbf{x})$ at these points, and move to the one producing the largest $f(\mathbf{x})$ (unless all have $f(\mathbf{x})^* < f(\mathbf{x}^n)$, in which case don't move!). A fancier variation on this idea would let the mean of the statistical distribution of step lengths vary in response to the previous step length having produced the best-of-ten solution.

Whichever technique you end up selecting, in all cases the performance will be affected by the choice of the starting guess $\mathbf{x}_0$. That may seem like just a minor nuisance, but there are

situations where secondary extrema exist in the search space, and the choice of a "bad" starting guess may quickly and efficiently land you... on a secondary extremum. **Global optimization** is the art and black magic of locating the tallest of all maxima in all of the search space, independently of how the search algorithm is initialized. There is **no** fast-and-easy way to achieve this. One straightforward avenue is known as **iterated hill climbing**, which looks like this:

```
Initialization: x** = 0
for j = 1, 2, ... do
    Initialization: x = x_j
    for k = 1, 2, ... do
        Construct: Δx
        x* = x + Δx
        δ = |f(x*) − f(x)|
        if δ ≥ε then
          x = x*
        else
          Local maximum found: x*
          exit
        endif
    enddo
    if x* > x** then x** = x*
enddo
Global maximum found: x**
```

Obviously this is just the same as plain hill climbing, except that an outer loop has been added, which allows multiple hill climbing runs from a set of distinct starting guesses, and the global maximum is just the largest $\mathbf{x}^*$ value found for the ensemble of hill climbing trials. The crux is now to decide when the outer iteration is to be stopped. Again, there is **no** universal termination criterion that can ensure that the global optimum has been located. Global optimization is just plain hard.

Chapter 10 in Press *et al.*'s *Numerical Recipes* provides one of the most illuminating discussion I have ever read on classical maximization/minimization algorithms. For an introduction to global optimization, focusing on **genetic algorithms**, see my NCAR Technical Note cited in the bibliography at the end of this Appendix. Press *et al.* covers very well **simulated annealing**, another powerful global optimization method.

F.5 Ordinary differential equations: initial-value problems

We focus here on a prototypical inital-value problem, described by an ordinary differential equation (ODE) assuming the general form:

$$\frac{\mathrm{d}f}{\mathrm{d}t} = g(t, f) \tag{F.27}$$

with a known initial condition. The first step towards a numerical solution is to discretize the independent variable t on a mesh, here assumed equidistant:

$$t_{k+1} = t_k + h \ , \qquad k = 0, 1, 2, ... \tag{F.28}$$

where the initial condition is specified at t_0, so that $f(x_0) \equiv f_0$. Note that the RHS of eq. (F.27) can in general be a nonlinear function, i.e., it depends explicitly on $f(t)$ as well as on t.

Euler's method is based in a Taylors series expansion truncated to first order, which is equivalent to using a forward finite difference to discretize the LHS of eq. (F.27). If the RHS is evaluated at t_k, then we have the algorithm known as Euler-explicit:

$$f_{k+1} = f_k + h\, g(t_k, f_k) + O(h) \ , \qquad k = 0, 1, 2, ... \tag{F.29}$$

This is an extremely simple algorithm, often sufficient for many applications, and easy to generalize to systems of coupled ODEs. On the downside, its discretization error is of order $O(h)$, which forces the use of a very tight mesh, and the resulting numerical solution often becomes polluted by the accumulation of roundoff errors.

If you understood the logic behind the derivation of finite difference formulae of high order, you likely anticipated that algorithms more accurate than Euler-explicit can be obtained by retaining more terms in the Taylor series development for $f(t)$, and, if the ODE is nonlinear, by also developing the RHS of eq. (F.27) as a series. A wide variery of such higher-order algorithms can be produced in this manner, but in most circumstances either one of the following two should suffice:

(1) **Heun's method:**

$$f_{k+1} = f_k + \frac{h}{2}\left[g(t_k, f_k) + g(t_{k+1}, f_k + h\, g(t_k, f_k))\right] + O(h^2) \ . \tag{F.30}$$

Note that this is similar to Euler-explicit, with the important difference that the RHS of eq. (F.30) is now calculated as an average of $g(t, f)$ evaluated at (t_k, f_k) and à t_{k+1} by linear extrapolation of f. This variation on a theme already brings the discretization error down to $O(h^2)$.

(2) **Fourth-order Runge Kutta method** (often just called Runge-Kutta); The idea here is to divide the time step h in substeps through which the solution is sequentially advanced, re-evaluating $g(t, f)$ at each sub-step:

$$f_{k+1} = f_k + \frac{h}{6}(G_1 + 2G_2 + 2G_3 + G_4) + O(h^4) \ , \tag{F.31}$$

where

$$G_1 = g(t_k, f_k) \ , \tag{F.32}$$

$$G_2 = g\left(t_k + h/2, f_k + (h/2)G_1\right) \ , \tag{F.33}$$

$$G_3 = g\left(t_k + h/2, f_k + (h/2)G_2\right) \ , \tag{F.34}$$

$$G_4 = g\left(t_{k+1}, f_k + hG_3\right) \ . \tag{F.35}$$

For all details regarding the design of these algorithms, see chapter 2 of the excellent book by Golub & Ortega, and/or chapter 16 of Press *et al.*, both cited in the bibliography at the end of this appendix. The monograph by Wood, also cited below, is a also a good, more specialized reference.

F.6 Eigenvalues and eigenvectors: inverse iteration

We consider here a generalized, discrete linear eigenvalue problem of the form:

$$\lambda[B]\mathbf{a} = [A]\mathbf{a} \ , \tag{F.36}$$

where the matrices $[A]$ and $[B]$ result from the application of finite difference formulae (for example), and λ and $\mathbf{a}$ are the sought-after eigenvalue and eigenvector. The matrices $[A]$

and $[B]$ are square, and of dimension equal to the number of nodes making up the mesh on which the (continuous) problem is being spatially discretized. In general λ and $\mathbf{a}$ are complex quantities. However, in some cases such as the diffusive decay problem of §7.1, the form of the differential operator and the use of centered finite differences, would lead $[B] = [I]$ and ensure that $[A]$ is real, tridiagonal and symmetric, which implies that the sought after eigenvalue λ and eigenvector $\mathbf{a}$ are real quantities (as one might have expected intuitively in the case of a purely diffusive problem). Note that since we have discretized the problem on a finite-size mesh, the infinite set of eigenvalues associated with the differential operator has effectively been truncated to a finite set of N eigenvalues/vectors, where N is the size of the matrices $[A]$ and $[B]$.

The technique known as **inverse iteration** provides an efficient and simple algorithm to solve eq. (F.36). The method requires as input a *trial* eigenvalue σ and trial eigenvector $\mathbf{a}^{(0)}$. The algorithm is as follows:

Initialize: σ, $\mathbf{a}^{(0)}$
Construct: $[C] = ([A] - \sigma[B])$
for k=1,2,...
  Solve: $[C]\mathbf{z}^{(k)} = [B]\mathbf{a}^{(k-1)}$
  Normalize: $\mathbf{a}^{(k)} = \mathbf{z}^{(k)}/\|\mathbf{z}^{(k)}\|_2$
  $\lambda^{(k)} = (\mathbf{a}^{(k)})^H[A]\mathbf{a}^{(k)}/(\mathbf{a}^{(k)})^H[B]\mathbf{a}^{(k)}$
endfor

where (k) is the iteration count, and $(\mathbf{a})^H$ is the Hermitian of $\mathbf{a}$ (i.e., the transpose of the complex conjugate). Note that since we are dealing with real matrices, eigenvalues and eigenvectors, we have $\mathbf{a}^H \equiv \mathbf{a}^T$, and and $\|\mathbf{z}\|_2 \equiv \mathbf{z} \cdot \mathbf{z}$. The iteration is stopped once the change in the eigenvalue from one step to the next becomes smaller than some pre-established tolerance criterion, for example when

$$\left| \frac{\lambda^{(k)} - \lambda^{(k-1)}}{\lambda^{(k)}} \right| \leq 10^{-6} \ . \tag{F.37}$$

At this point, both a solution eigenvector $\mathbf{a}$ and a improved eigenvalue $\lambda \neq \sigma$ have been produced. The algorithm does require a reasonable guess for the trial eigenvalue σ, but is largely insensitive to the initial guess $\mathbf{a}^{(0)}$ for the eigenvector, as long as it satisfies the boundary conditions. Operationally, an attractive feature of inverse iteration is the fact that the matrix $[C]$ needs only be constructed once and for all using only the trial eigenvalue; it can then be triangularized once (a N^3 operation), with each iteration then requiring only a matrix-vector multiplication and a RHS reduction (both being N^2 operations) on a RHS vector that is recomputed from one iteration to the next. See §7.6 of the Golub & Van Loan textbook cited below for further details.

Inverse iteration is a very useful technique, and you should make an effort to code it up and become familiar with it. By now most scientific programmation languages include canned packages and/or libraries for the solution of linear systems of equations in matrix form, so your coding effort will actually be minimal. You may reflect upon the fact that the the diffusive decay problem of §7.1, The Roberts Cell dynamo solution of §8.1, as well as the linear mean-field dynamo models of 10.2.2, were all computed by inverse iteration.

In general, the system can be expected to have many distinct eigenvalues; inverse iteration will almost always converge to the eigenvalue closest (in the complex plane) to the trial eigenvalue σ. The only way that inverse iteration, used as described above, can produce all eigenvalues and eigenvectors that satisfy eq. (F.36) is by trial and error search, namely by starting the iteration with different values of σ. This is not a very efficient way to proceed, but then again in practice one is not always interested in finding *all* eigenvalues and eigenmodes of the system. If all eigenvalues/vectors are required, or if no suitable guesses at the eigenvalues can be formulated, then the procedure to follow would be to use an algorithm such as

QR decomposition to obtain moderately accurate eigenvalues, and then use inverse iteration to produce more accurate eigenvalues *and* their corresponding eigenvectors. Those of you already familiar with the numerical treatment of eigenvalue problems may want to try to approach eigenproblems in this more satisfactory manner.

Bibliography:

What follows is a simple alphabetical listing of the books cited in this appendix. Note that three versions of the Press *et al.* book exist; they differ mainly (but not only) in the programming language used for the code listings provided in the book (you can choose from FORTRAN, PASCAL and C).

Charbonneau, P. 2002, *An introduction to genetic algorithms for numerical optimization*, NCAR Technical Note TN-450+IA (Boulder: National Center for Atmospheric Research.

Gerald, C.F. 1978, *Applied numerical analysis*, Addison-Wesley.

Golub, G. H., & Ortega, J. M. 1992, *Scientific computing and differential equations*, Academic Press.

Golub, G. H., & Van Loan, C. F. 1989, *Matrix Computations*, second ed., Johns Hopkins University Press.

Lapidus, L., & Pinder, G.F. 1982, *Numerical solution of partial differential equations in science and engineering*, John Wiley & Sons.

Press, W.H., Teukolsky, S.A., Vetterling, W.T., et Flannery, B.F., *Numerical Recipes*, Cambridge University Press (2^e édition 1992).

Wood, W. L., *Practical time-stepping schemes*, Clarendon Press (1990).