Geodynamo Boundary Conditions with Topography in Poloidal–Toroidal Form

Overview

We consider a Boussinesq, electrically conducting fluid outer core between radii $r_i \le r \le r_o$, bounded by a solid inner core (IC) and an insulating mantle (M). Both the inner-core boundary (ICB) and the core–mantle boundary (CMB) may have small topography,

$$r = r_b + \varepsilon\, h_b(\theta, \phi), \qquad b \in \{i, o\}, \quad 0 < \varepsilon \ll 1. \tag{1}$$

Surface (tangential) differential operators are denoted $\boldsymbol{\nabla}_H$ and the scalar surface Laplacian operator is $L^2 \equiv -\, r^2\, \boldsymbol{\nabla}_H \cdot \boldsymbol{\nabla}_H$, so that $L^2 Y_\ell^m = \ell(\ell+1) Y_\ell^m$.

We present implementable boundary conditions and their spherical-harmonic projections for:

- Velocity $\boldsymbol{u}$ (impermeability, no-slip or stress-free),
- Magnetic field $\boldsymbol{B}$ (CMB insulating, ICB insulating or conducting),
- Temperature T (Dirichlet: fixed T, Neumann: fixed flux),
- Stefan condition at the ICB (phase change) and topography evolution $h_i(t)$.

1 Poloidal–Toroidal decomposition

Decompose the solenoidal fields

$$\boldsymbol{u} = \boldsymbol{\nabla} \times (T_u\, \hat{\boldsymbol{r}}) \;+\; \boldsymbol{\nabla} \times \boldsymbol{\nabla} \times (P_u\, \hat{\boldsymbol{r}}), \tag{2}$$

$$\boldsymbol{B} = \boldsymbol{\nabla} \times (T_b\, \hat{\boldsymbol{r}}) \;+\; \boldsymbol{\nabla} \times \boldsymbol{\nabla} \times (P_b\, \hat{\boldsymbol{r}}). \tag{3}$$

Useful component forms on a sphere of radius r are

$$u_r = \frac{L^2 P_u}{r^2}, \qquad \boldsymbol{u}_t = \frac{1}{r}\, \partial_r(\boldsymbol{\nabla}_H P_u) + \frac{1}{r}\, \hat{\boldsymbol{r}} \times \boldsymbol{\nabla}_H T_u, \tag{4}$$

$$B_r = \frac{L^2 P_b}{r^2}, \qquad \boldsymbol{B}_t = \frac{1}{r}\, \partial_r(\boldsymbol{\nabla}_H P_b) + \frac{1}{r}\, \hat{\boldsymbol{r}} \times \boldsymbol{\nabla}_H T_b. \tag{5}$$

Expand the scalars as $P_\star(r, \theta, \phi) = \sum_{\ell m} P_{\star,\ell m}(r) Y_\ell^m$ and similarly for $T_\star$.

2 Geometry and linearization with topography

The outward unit normal and normal derivative on the true surface, linearized at $r = r_b$, are

$$\hat{\boldsymbol{n}}_b = \hat{\boldsymbol{r}} - \frac{\varepsilon}{r_b}\,\boldsymbol{\nabla}_H h_b, \tag{6}$$

$$\partial_n \approx \partial_r - \frac{1}{r_b}\,\boldsymbol{\nabla}_H h_b\cdot\boldsymbol{\nabla}_H + h_b\,\partial_{rr}. \tag{7}$$

The liquid-side normal velocity evaluated on the reference sphere r_b is

$$u_n \approx u_r - \frac{1}{r_b}\,\boldsymbol{\nabla}_H h_b\cdot\boldsymbol{u}_t + h_b\,\partial_r u_r. \tag{8}$$

The leading couplings to topography are the 'slope' (proportional to $\boldsymbol{\nabla}_H h_b$) and the 'shift' (proportional to h_b) terms.

3 Mechanical boundary conditions

Impermeability on the true surface (kinematic), linearized:

$$\boldsymbol{u}\cdot\hat{\boldsymbol{n}}_b = 0 \;\Rightarrow\; u_r - \frac{1}{r_b}\,\boldsymbol{\nabla}_H h_b\cdot\boldsymbol{u}_t + h_b\,\partial_r u_r = 0 \quad (r = r_b). \tag{9}$$

Tangential conditions (choose one):

No-slip: $\boldsymbol{u}_t(r_b)+h_b\,\partial_r\boldsymbol{u}_t(r_b) = \boldsymbol{U}_{b,t}$ where typically $\boldsymbol{U}_{o,t} = \boldsymbol{0}$ for the mantle, and $\boldsymbol{U}_{i,t} = (\boldsymbol{\Omega}_{ic}\times\boldsymbol{r})_t$ at the ICB.

Stress-free: vanishing tangential viscous traction on the true surface gives, to first order,

$$\partial_r\left(\frac{\boldsymbol{u}_t}{r}\right)\Big|_{r_b} = \frac{1}{r_b^2}\,(\boldsymbol{\nabla}_H h_b)\,\partial_r u_r\Big|_{r_b}, \tag{10}$$

commonly implemented in the flat limit as $\partial_r(\boldsymbol{u}_t/r) = 0$ at r_b.

4 Magnetic boundary conditions

CMB with insulating mantle

Outside the core $\boldsymbol{B} = -\boldsymbol{\nabla}\Phi$, $\nabla^2\Phi = 0$, thus $\Phi = \sum_{\ell m} a_{\ell m}\, r^{-(\ell+1)} Y_\ell^m$.

- Toroidal: $T_b(r_o) + h_o\,\partial_r T_b(r_o) = 0$.
- Poloidal: match the tangential field (or potential) on the true surface. In the flat limit

$$\partial_r P_{b,\ell m}(r_o) + \frac{\ell+1}{r_o} P_{b,\ell m}(r_o) = 0, \qquad T_{b,\ell m}(r_o) = 0, \tag{11}$$

and topography introduces linear couplings among (ℓ, m) via Gaunt-type tensors (see Section 7).

ICB with insulating inner core

The solid interior is potential and regular at the origin: $\Phi^{\text{int}} = \sum_{\ell m} b_{\ell m}\, r^\ell Y_\ell^m$.

$$T_b(r_i) + h_i\,\partial_r T_b(r_i) = 0,$$
$$\partial_r P_{b,\ell m}(r_i) - \frac{\ell}{r_i} P_{b,\ell m}(r_i) + (\text{topography couplings}) = 0. \tag{12}$$

ICB with conducting inner core

Let $\eta = 1/(\mu_0\sigma)$ and $\eta_{ic} = 1/(\mu_0\sigma_{ic})$ be magnetic diffusivities. On the true surface (linearized at r_i) enforce

$$\text{Continuity of } B_r \text{ and } \boldsymbol{B}_t, \tag{13}$$
$$\text{Continuity of } \boldsymbol{E}_t: \quad \big[-\boldsymbol{u}\times\boldsymbol{B} + \eta\,(\boldsymbol{\nabla}\times\boldsymbol{B})\big]_t^{\text{fluid}} = \big[-\boldsymbol{U}_{ic}\times\boldsymbol{B} + \eta_{ic}\,(\boldsymbol{\nabla}\times\boldsymbol{B})\big]_t^{\text{IC}}. \tag{14}$$

In spherical harmonics these give two scalar relations per (ℓ, m) coupling P_b, T_b and their radial derivatives on both sides (plus topographic couplings).

5 Temperature align and thermal BCs

Write $T(r,\theta,\phi,t) = T_{\text{cond}}(r) + \Theta(r,\theta,\phi,t)$. The perturbation temperature satisfies

$$\partial_t\Theta + \boldsymbol{u}\cdot\boldsymbol{\nabla}\Theta + u_r\,\partial_r T_{\text{cond}} = \kappa\nabla^2\Theta + \frac{H}{\rho c_p}. \tag{15}$$

Thermal boundary conditions on the true surface, linearized at $r = r_b$:

Dirichlet (fixed T):

$$\Theta(r_b) + h_b\,\partial_r\Theta(r_b) = T_b(\theta,\phi,t) - T_{\text{cond}}(r_b) - h_b\,\partial_r T_{\text{cond}}(r_b). \tag{16}$$

Neumann (fixed flux $-k\,\partial_n T = q_b$):

$$-k\left[\partial_r\Theta - \frac{1}{r_b}\boldsymbol{\nabla}_H h_b\cdot\boldsymbol{\nabla}_H\Theta + h_b\,\partial_{rr}\Theta\right] = q_b + k\left[\partial_r T_{\text{cond}} - h_b\,\partial_{rr}T_{\text{cond}}\right]. \tag{17}$$

6 Stefan condition and ICB topography evolution

At the moving solid–liquid interface (ICB), with unit normal into the solid and interface speed $V_b = \partial_t h_i$:

$$k_{ic}\,\partial_n T_{ic} - k\,\partial_n T = \rho\,L\,(V_b - u_n). \tag{18}$$

Equivalently, an *implementable* update for the topography is

$$\boxed{\partial_t h_i = u_n + \frac{k_{ic}\,\partial_n T_{ic} - k\,\partial_n T}{\rho\,L}}. \tag{19}$$

Using the linearized operators at r_i,

$$\partial_n \approx \partial_r - \frac{1}{r_i}\boldsymbol{\nabla}_H h_i\cdot\boldsymbol{\nabla}_H + h_i\,\partial_{rr},$$
$$u_n \approx u_r - \frac{1}{r_i}\boldsymbol{\nabla}_H h_i\cdot\boldsymbol{u}_t + h_i\,\partial_r u_r. \tag{20}$$

Temperature continuity at the ICB is typically imposed ($T = T_{ic} = T_m$ on the true surface, optionally with Clapeyron/compositional corrections).

7 Spectral projection and Gaunt couplings

Let $\Theta = \sum_{\ell m} \Theta_{\ell m}(r) Y_\ell^m$, $\Theta_{ic} = \sum_{\ell m} \Theta_{\ell m}^{ic}(r) Y_\ell^m$ and $h_b = \sum_{LM} h_{LM}^b Y_L^M$. Define the Gaunt-type tensors

$$\mathcal{G}_{\ell m,\ell' m',LM} = \int Y_\ell^{m*}\, Y_{\ell'}^{m'}\, Y_L^M \, \mathrm{d}\Omega, \tag{21}$$

$$\mathcal{G}^{(\nabla)}_{\ell m,\ell' m',LM} = \int Y_\ell^{m*}\, \boldsymbol{\nabla}_H Y_{\ell'}^{m'} \cdot \boldsymbol{\nabla}_H Y_L^M \, \mathrm{d}\Omega, \tag{22}$$

$$\mathcal{G}^{(\times)}_{\ell m,\ell' m',LM} = \int Y_\ell^{m*}\, \hat{\boldsymbol{r}} \cdot (\boldsymbol{\nabla}_H Y_{\ell'}^{m'} \times \boldsymbol{\nabla}_H Y_L^M) \, \mathrm{d}\Omega. \tag{23}$$

Examples of projected boundary relations:

- **Impermeability** (schematic):

$$\begin{aligned}
&\frac{\ell(\ell+1)}{r_b^2} P_{u,\ell m} + \sum_{LM\ell' m'} h_{LM}^b \, \mathcal{G}_{\ell m,\ell' m',LM} \, \partial_r \left(\frac{\ell'(\ell'+1)}{r^2} P_{u,\ell' m'} \right) \\
&- \frac{1}{r_b^2} \sum_{LM\ell' m'} h_{LM}^b \left[\partial_r P_{u,\ell' m'} \, \mathcal{G}^{(\nabla)}_{\ell m,\ell' m',LM} + T_{u,\ell' m'} \, \mathcal{G}^{(\times)}_{\ell m,\ell' m',LM} \right] = 0.
\end{aligned} \tag{24}$$

- **CMB insulating** (schematic):

$$\begin{aligned}
\left[\partial_r P_{b,\ell m} + \frac{\ell+1}{r_o} P_{b,\ell m} \right] + \sum_{LM\ell' m'} h_{LM}^o \left(\alpha^o \, \partial_r P_{b,\ell' m'} + \beta^o \, T_{b,\ell' m'} + \gamma^o \, P_{b,\ell' m'} \right) &= 0, \\
T_{b,\ell m}(r_o) &= 0.
\end{aligned} \tag{25}$$

8 Thermal projections and Stefan (matrix form)

Dirichlet at r_b:

$$\Theta_{\ell m} + \sum_{LM\ell' m'} h_{LM}^b \, \mathcal{G}_{\ell m,\ell' m',LM} \, \partial_r \Theta_{\ell' m'} = \left[T_b - T_{\mathrm{cond}}(r_b) \right]_{\ell m} - \sum_{LM} h_{LM}^b \, \mathcal{G}_{\ell m,00,LM} \, \partial_r T_{\mathrm{cond}}. \tag{26}$$

Neumann at r_b (you may omit the $\partial_{rr}\Theta$ shift term at first pass):

$$\begin{aligned}
\partial_r \Theta_{\ell m} - \frac{1}{r_b} \sum_{LM\ell' m'} h_{LM}^b \, \mathcal{G}^{(\nabla)}_{\ell m,\ell' m',LM} \, \Theta_{\ell' m'} + \sum_{LM\ell' m'} h_{LM}^b \, \mathcal{G}_{\ell m,\ell' m',LM} \, \partial_{rr} \Theta_{\ell' m'} \\
= -\frac{q_{b,\ell m}}{k} - \partial_r T_{\mathrm{cond}} \, \delta_{\ell 0} \delta_{m0} + \sum_{LM} h_{LM}^b \, \mathcal{G}_{\ell m,00,LM} \, \partial_{rr} T_{\mathrm{cond}}.
\end{aligned} \tag{27}$$

ICB Stefan update (projected, schematic):

$$\partial_t h_{\ell m}^i = u_{n,\ell m} + \frac{1}{\rho L} \, \mathcal{F}_{\ell m}, \tag{28}$$

where $u_{n,\ell m}$ is built from P_u, T_u using $\mathcal{G}^{(\nabla)}$ and $\mathcal{G}^{(\times)}$, and $\mathcal{F}_{\ell m}$ collects $k_{ic}\partial_r \Theta^{ic} - k\partial_r \Theta$ plus optional slope/shift corrections $\propto h_i$.

9 Dimensionless form (quick reference)

With length D, time D^2/κ (liquid), define $\lambda = k_{ic}/k$ and the Stefan number $\mathrm{Ste} = c_p \Delta T/L$. Then

$$\partial_t h_i^* = u_n^* + \frac{1}{\mathrm{Ste}}\Big(\lambda\, \partial_n \Theta_{ic}^* - \partial_n \Theta^*\Big), \tag{29}$$

with the same topography-linearized operators for ∂_n and u_n as above. Flat magnetic checks:

$$\begin{aligned} \text{CMB:} \quad & \partial_r P_{b,\ell m} + \frac{\ell+1}{r_o} P_{b,\ell m} = 0, \quad T_{b,\ell m} = 0, \\ \text{ICB (insulating):} \quad & \partial_r P_{b,\ell m} - \frac{\ell}{r_i} P_{b,\ell m} = 0, \quad T_{b,\ell m} = 0. \end{aligned} \tag{30}$$

Implementation checklist.

1. Precompute $\mathcal{G}$, $\mathcal{G}^{(\nabla)}$, $\mathcal{G}^{(\times)}$ up to $L_{\max}$.
2. Assemble flat-sphere boundary operators (mechanical, magnetic, thermal).
3. Add topographic coupling blocks proportional to h_b (slope first; add shift terms later).
4. If evolving the ICB: impose temperature continuity, apply Stefan semi-implicitly, update h_i in spectral space.
5. Validate against flat limits and standard benchmarks before enabling all couplings.

Appendix A: Example boundary operator blocks (single ℓ)

This appendix shows compact, matrix-friendly boundary rows for a *single* spherical-harmonic (ℓ, m) in the *flat* limit with first-order topographic couplings indicated symbolically. Let $r_b \in \{r_i, r_o\}$. We denote by $\mathsf{row}[\cdot]$ a linear combination of the boundary unknowns evaluated at r_b.

A.1 Impermeability $(u_n = 0)$ at $r = r_b$

For $\ell \geq 1$,

$$\mathsf{row}_{\mathrm{imp}}: \quad \underbrace{\frac{\ell(\ell+1)}{r_b^2} P_{u,\ell m}}_{\text{flat}} + \sum_{\ell' m' LM} h_{LM}^b \Big[\underbrace{\alpha_{\ell m,\ell' m',LM}^{\mathrm{imp}}\, \partial_r P_{u,\ell' m'} + \beta_{\ell m,\ell' m',LM}^{\mathrm{imp}}\, T_{u,\ell' m'}}_{\text{slope terms}} + \underbrace{\gamma_{\ell m,\ell' m',LM}^{\mathrm{imp}}\, P_{u,\ell' m'}}_{\text{shift}} \Big] = 0. \tag{31}$$

The coefficients are contractions of Gaunt tensors from Sec. 7; for example (schematic) $\alpha^{\mathrm{imp}} \propto r_b^{-2}\, \mathcal{G}^{(\nabla)}$, $\beta^{\mathrm{imp}} \propto r_b^{-2}\, \mathcal{G}^{(\times)}$, and $\gamma^{\mathrm{imp}} \propto \partial_r\big[\ell(\ell+1) P_u / r^2\big]$ projected.

A.2 Stress-free at $r = r_b$

A practical implementation uses the flat-sphere form $\partial_r(\boldsymbol{u}_t/r) = 0$ and adds h_b-couplers:

$$\mathsf{row}_{\mathrm{sf}}: \quad \underbrace{\frac{\ell(\ell+1)}{r_b^3}\big[-P_{u,\ell m}\big] + \frac{\ell(\ell+1)}{r_b^2}\,\partial_r P_{u,\ell m} + \frac{1}{r_b^2}\,\partial_r T_{u,\ell m}}_{\text{flat, typical}} + \sum h^b_{LM}\,(\cdots) = 0. \tag{32}$$

(The exact prefactors depend on your toroidal/poloidal normalization; treat them as templates to code from.)

A.3 CMB magnetic (insulating)

$$\begin{aligned} \mathsf{row}_{\mathrm{CMB}}: \quad & \big[\partial_r P_{b,\ell m} + (\ell+1)P_{b,\ell m}/r_o\big] + \sum h^o_{LM}\left(\alpha^o\,\partial_r P_{b,\ell' m'} + \beta^o\,T_{b,\ell' m'} + \gamma^o\,P_{b,\ell' m'}\right) = 0, \\ & \mathsf{row}_T: \; T_{b,\ell m} = 0 \text{ (or } T_b + h_o\partial_r T_b = 0). \end{aligned} \tag{33}$$

A.4 ICB magnetic (insulating)

$$\begin{aligned} \mathsf{row}_{\mathrm{ICB}}: \quad & \big[\partial_r P_{b,\ell m} - \ell\,P_{b,\ell m}/r_i\big] + \sum h^i_{LM}\,(\alpha^i\,\partial_r P + \beta^i\,T + \gamma^i\,P) = 0, \\ & \mathsf{row}_T: \; T_{b,\ell m} = 0 \text{ (or } T_b + h_i\partial_r T_b = 0). \end{aligned} \tag{34}$$

A.5 Thermal BCs

Dirichlet (*value* of T):

$$\mathsf{row}^{\mathrm{Dir}}_{\Theta}: \quad \Theta_{\ell m} + \sum h^b_{LM}\,\mathcal{G}_{\ell m,\ell' m',LM}\,\partial_r\Theta_{\ell' m'} = \big[T_b - T_{\mathrm{cond}}(r_b)\big]_{\ell m} - \sum h^b_{LM}\,\mathcal{G}_{\ell m,00,LM}\,\partial_r T_{\mathrm{cond}}. \tag{35}$$

Neumann (*flux*):

$$\begin{aligned} \mathsf{row}^{\mathrm{Neu}}_{\Theta}: \quad & \partial_r\Theta_{\ell m} - \frac{1}{r_b}\sum h^b_{LM}\,\mathcal{G}^{(\nabla)}_{\ell m,\ell' m',LM}\,\Theta_{\ell' m'} + \sum h^b_{LM}\,\mathcal{G}_{\ell m,\ell' m',LM}\,\partial_{rr}\Theta_{\ell' m'} \\ & = -\frac{q_{b,\ell m}}{k} - \partial_r T_{\mathrm{cond}}\,\delta_{\ell 0}\delta_{m0} + \sum h^b_{LM}\,\mathcal{G}_{\ell m,00,LM}\,\partial_{rr} T_{\mathrm{cond}}. \end{aligned} \tag{36}$$

A.6 Stefan update (semi-implicit, per mode)

With timestep Δt, update $h^i_{\ell m}$ as

$$h^{n+1}_{\ell m} = h^n_{\ell m} + \Delta t\left[u^n_{n,\ell m} + \frac{1}{\rho L}\left(k_{ic}\,\partial_n\Theta^{ic,n+1}_{\ell m} - k\,\partial_n\Theta^{n+1}_{\ell m}\right)\right], \tag{37}$$

using $\partial_n = \partial_r - (1/r_i)\sum h^i_{LM}\mathcal{G}^{(\nabla)}(\cdot) + \ldots$ built from the current h^i. This treats the fluxes implicitly and the kinematics explicitly (robust and simple).

Appendix B: Minimal Gaunt tensor routines (Python/Julia/Fortran)

Below are compact routines to evaluate the basic Gaunt integral $\mathcal{G} = \int Y_{\ell_1}^{m_1} Y_{\ell_2}^{m_2} Y_{\ell_3}^{m_3} \, \mathrm{d}\Omega$ and the gradient/cross variants numerically on a tensor-product grid. They do not require special libraries beyond standard FFT/linear algebra. For higher accuracy/speed, replace the numerical quadrature by analytic $3j$ symbols.

B.1 Python (NumPy)

```python
import numpy as np
from numpy.polynomial.legendre import leggauss
try:
    from scipy.special import sph_harm, wigner_3j   # optional
    HAVE_SCIPY = True
except Exception:
    HAVE_SCIPY = False

def sphY(l, m, theta, phi):
    if HAVE_SCIPY:
        return sph_harm(m, l, phi, theta)   # SciPy uses (m,l,phi,theta)
    # minimal fallback: real SH via numpy; for production prefer SciPy
    from math import factorial
    # naive (slow) assoc. Legendre via recursion could be added here
    raise RuntimeError("Install␣SciPy␣for␣sph_harm␣or␣plug␣your␣own␣Y_lm.")

def gauss_legendre_sphere(nth=96, nphi=192):
    mu, w = leggauss(nth)                # mu in [-1,1], weights sum to 2
    theta = np.arccos(mu)                # [0,pi]
    phi = np.linspace(0, 2*np.pi, nphi, endpoint=False)
    wphi = (2*np.pi)/nphi * np.ones_like(phi)
    return theta, phi, w, wphi

def gaunt(l1,m1,l2,m2,l3,m3, nth=96, nphi=192):
    # Analytic shortcut if SciPy Wigner-3j is available:
    if HAVE_SCIPY:
        f = np.sqrt((2*l1+1)*(2*l2+1)*(2*l3+1)/(4*np.pi))
        w3 = wigner_3j(l1,l2,l3,0,0,0) * wigner_3j(l1,l2,l3,m1,m2,m3)
        return float(f * w3)
    # Otherwise do numerical quadrature
    th, ph, wth, wph = gauss_legendre_sphere(nth, nphi)
    val = 0.0j
    for i, (t, wt) in enumerate(zip(th, wth)):
        Y1 = sphY(l1,m1,t,0)  # phi phase split out below
        Y2 = sphY(l2,m2,t,0)
        Y3 = sphY(l3,m3,t,0)
        # accumulate over phi analytically using e^{i m phi} orthogonality:
        if m1+m2+m3 != 0:
            continue
        val += wt * 2*np.pi * (Y1*Y2*Y3).real
    return float(val)

def grad_surf_components(Y, th, ph):
    # Compute surface gradient components (theta,phi) by finite differences
    # d/dphi is exact: dphi Y = i m Y for complex SH, but we FD for generality
    dth = th[1]-th[0]
```

```
    Yt = np.gradient(Y, dth, axis=0, edge_order=2)  # approx d/dtheta
    dphi = ph[1]-ph[0]
    Yp = np.gradient(Y, dphi, axis=1, edge_order=2) # approx d/dphi
    return Yt, Yp

def gaunt_grad_dot(l1,m1,l2,m2,l3,m3, nth=96, nphi=192):
    th, ph, wth, wph = gauss_legendre_sphere(nth, nphi)
    Th, Ph = np.meshgrid(th, ph, indexing='ij')
    Y1 = sphY(l1,m1,Th,Ph); Y2 = sphY(l2,m2,Th,Ph); Y3 = sphY(l3,m3,Th,Ph)
    Y2t, Y2p = grad_surf_components(Y2, th, ph)
    Y3t, Y3p = grad_surf_components(Y3, th, ph)
    # grad_H Y = theta_hat * dtheta Y + phi_hat * (1/sin) dphi Y
    sinth = np.sin(Th)
    dot23 = (Y2t * Y3t + (Y2p * Y3p) / (sinth**2))
    integrand = np.real(np.conj(Y1) * dot23) * sinth
    return float(np.tensordot(wth, np.tensordot(integrand, wph, axes=([1],[0])), axe

def gaunt_cross(l1,m1,l2,m2,l3,m3, nth=96, nphi=192):
    th, ph, wth, wph = gauss_legendre_sphere(nth, nphi)
    Th, Ph = np.meshgrid(th, ph, indexing='ij')
    Y1 = sphY(l1,m1,Th,Ph); Y2 = sphY(l2,m2,Th,Ph); Y3 = sphY(l3,m3,Th,Ph)
    Y2t, Y2p = grad_surf_components(Y2, th, ph)
    Y3t, Y3p = grad_surf_components(Y3, th, ph)
    sinth = np.sin(Th)
    # rhat\cdot(grad Y2 \times grad Y3) = (1/sin\theta)( d\theta Y2 d\phi Y3 - d\ph
    cross23 = (Y2t*Y3p - Y2p*Y3t) / sinth
    integrand = np.real(np.conj(Y1) * cross23) * sinth
    return float(np.tensordot(wth, np.tensordot(integrand, wph, axes=([1],[0])), axe
```

B.2 Julia (skeleton)

```
using FastGaussQuadrature, SpecialFunctions

function gaunt(l1,m1,l2,m2,l3,m3; nth=96, nphi=192)
    # If Wigner3j is available, prefer analytic formula.
    # Otherwise, numerical quadrature skeleton:
      , w   = gausslegendre(nth)   #        [-1,1]
       = acos.( )
       = range(0, 2  ; length=nphi+1)[1:end-1]
     w   = fill(2  /nphi, nphi)
    # TODO: implement Y_lm(  ,  ) or use SphericalHarmonics.jl
    return 0.0
end
```

B.3 Fortran (skeleton)

```
! Minimal skeleton using Gauss-Legendre in theta and uniform phi.
! Provide your Y_lm and its theta/phi derivatives, then integrate.
function gaunt(l1,m1,l2,m2,l3,m3, nth, nphi) result(val)
  implicit none
  integer, intent(in) :: l1,m1,l2,m2,l3,m3,nth,nphi
  real*8 :: val
  ! ... allocate grids, call your Ylm(  ,  ), accumulate integral ...
  val = 0d0
end function gaunt
```

B.4 Identities (optional speedups)

When analytic $3j$ symbols are available,

$$\mathcal{G}_{123} = \sqrt{\frac{(2\ell_1+1)(2\ell_2+1)(2\ell_3+1)}{4\pi}} \begin{pmatrix} \ell_1 & \ell_2 & \ell_3 \\ 0 & 0 & 0 \end{pmatrix} \begin{pmatrix} \ell_1 & \ell_2 & \ell_3 \\ m_1 & m_2 & m_3 \end{pmatrix}. \tag{38}$$

Moreover,

$$\int (\nabla_H Y_2)\cdot(\nabla_H Y_3)\, Y_1^* \,\mathrm{d}\Omega = \tfrac{1}{2}\left[\ell_2(\ell_2+1) + \ell_3(\ell_3+1) - \ell_1(\ell_1+1)\right] \mathcal{G}_{123}, \tag{39}$$

which avoids computing surface gradients explicitly. The $\mathcal{G}^{(\times)}$ integral can be evaluated numerically via the formula used in `gaunt_cross` above.

Appendix C: Tiny assembly demo (pseudo-code)

Below is a compact sketch (Python-like) of assembling boundary rows for a given (ℓ, m):

```
# Unknowns at boundary r_b for this (  ,m):
# x = [Pu, dPu, Tu, dTu, Pb, dPb, Tb, dTb, Th, dTh, Th_ic, dTh_ic]

rows = []

# Impermeability (flat + topography couplers)
A0 = (ell*(ell+1)/rb**2)   # multiplies Pu
row_imp = {'Pu': A0}
for (L,M), hLM in topography.items():
    # slope terms with Gaunt gradients:
    row_imp[('dPu',L,M)] = hLM * alpha_imp(ell,m,L,M)
    row_imp[('Tu',L,M)]  = hLM * beta_imp(ell,m,L,M)
    # optional shift term:
    row_imp[('Pu',L,M)]  = hLM * gamma_imp(ell,m,L,M)
rows.append(row_imp)

# Magnetic CMB insulating: dPb + (   +1)Pb/r_o + couplers = 0 ; Tb = 0
row_cmb = {'dPb': 1.0, 'Pb': (ell+1)/ro}
for (L,M), hLM in topography_o.items():
    row_cmb[('dPb',L,M)] += hLM * alpha_o(ell,m,L,M)
    row_cmb[('Tb',L,M)]  += hLM * beta_o(ell,m,L,M)
    row_cmb[('Pb',L,M)]  += hLM * gamma_o(ell,m,L,M)
rows.append(row_cmb)
rows.append({'Tb': 1.0})  # flat: Tb=0

# Thermal Dirichlet at r_b (example):    +    h G    r    = RHS
row_Th = {'Th': 1.0}
for (L,M), hLM in topography.items():
    row_Th[('dTh',L,M)] = hLM * G(ell,m,L,M)   # Gaunt coefficient
rows.append(row_Th)

# Stefan update (semi-implicit): h^{n+1} = h^n + dt*(u_n^n +
(k_ic d_n Th_ic^{n+1} - k d_n Th^{n+1})/(\rho L))
# Put flux part on left-hand side along with your temperature unknowns; update h se
```