Chapter 1

Geometric model of the semi-toroidal surface and rigorous validation

1.1 Goals of the chapter

This chapter provides a mathematically rigorous presentation of the semi-toroidal parameterization used in this work, derives the first and second fundamental forms, computes the unit normal, the coefficients of the second fundamental form, the principal curvatures, Gaussian and mean curvature; then formulates and proves error estimates for the chosen numerical quadrature schemes (composite midpoint, trapezoidal, Simpson), and finally presents a reproducible algorithm with a posteriori error bounds for the specific integrands arising in our model.

1.2 Parameterization and parameter domain

Let

$$\mathbf{X} : U \subset \mathbb{R}^2 \to \mathbb{R}^3, \qquad \mathbf{X}(u,v) = \begin{pmatrix} (R + r\cos v)\cos u \\ (R + r\cos v)\sin u \\ r \sin v \end{pmatrix},$$

with parameters $u \in [0, 2\pi]$, $v \in [0, \pi]$ and constants $R > r > 0$. Denote $U = [0, 2\pi] \times [0, \pi]$. The mapping $\mathbf{X}$ is C^∞ on U and regular (rank of Jacobian is 2) for all $(u, v) \in U$, as will be shown by explicit computation of the cross product $\mathbf{X}_u \times \mathbf{X}_v$.

1.3 First fundamental form (metric coefficients)

Define the partial derivatives

$$\mathbf{X}_u = \frac{\partial \mathbf{X}}{\partial u}, \qquad \mathbf{X}_v = \frac{\partial \mathbf{X}}{\partial v}.$$

The coefficients of the first fundamental form are

$$E = \langle \mathbf{X}_u, \mathbf{X}_u \rangle, \qquad F = \langle \mathbf{X}_u, \mathbf{X}_v \rangle, \qquad G = \langle \mathbf{X}_v, \mathbf{X}_v \rangle,$$

so that the first fundamental form (the induced metric) is

$$I = E\, du^2 + 2F\, du\, dv + G\, dv^2.$$

Computation for the semi-torus

Compute the partial derivatives explicitly:

$$\mathbf{X}_u = \begin{pmatrix} -(R + r\cos v)\sin u \\ (R + r\cos v)\cos u \\ 0 \end{pmatrix}, \qquad \mathbf{X}_v = \begin{pmatrix} -r\sin v\cos u \\ -r\sin v\sin u \\ r\cos v \end{pmatrix}.$$

Therefore

$$E = \mathbf{X}_u \cdot \mathbf{X}_u = (R + r\cos v)^2,$$
$$F = \mathbf{X}_u \cdot \mathbf{X}_v = 0,$$
$$G = \mathbf{X}_v \cdot \mathbf{X}_v = r^2.$$

Hence the metric is diagonal in parameters (u, v):

$$I = (R + r\cos v)^2\, du^2 + r^2\, dv^2.$$

The determinant of the metric tensor is

$$W = EG - F^2 = (R + r\cos v)^2\, r^2 > 0,$$

which confirms regularity for $R > r > 0$.

1.4 Unit normal and surface element

The oriented surface element is obtained from the cross product

$$\mathbf{X}_u \times \mathbf{X}_v.$$

Compute the cross product (algebraic expansion omitted here; see Appendix for step-by-step):

$$\mathbf{X}_u \times \mathbf{X}_v = r(R + r\cos v) \begin{pmatrix} \cos u\cos v \\ \sin u\cos v \\ \sin v \end{pmatrix}.$$

Its magnitude is

$$\|\mathbf{X}_u \times \mathbf{X}_v\| = r(R + r\cos v).$$

Therefore the unit normal (with the chosen orientation) is

$$\mathbf{n} = \frac{\mathbf{X}_u \times \mathbf{X}_v}{\|\mathbf{X}_u \times \mathbf{X}_v\|} = \begin{pmatrix} \cos u\cos v \\ \sin u\cos v \\ \sin v \end{pmatrix}.$$

The scalar surface element is

$$dS = \|\mathbf{X}_u \times \mathbf{X}_v\|\, du\, dv = r(R + r\cos v)\, du\, dv.$$

1.5 Second fundamental form and full derivation of coefficients

The second fundamental form coefficients are given by

$$e = \langle \mathbf{X}_{uu}, \mathbf{n} \rangle, \qquad f = \langle \mathbf{X}_{uv}, \mathbf{n} \rangle, \qquad g = \langle \mathbf{X}_{vv}, \mathbf{n} \rangle,$$

where $\mathbf{X}_{uu}, \mathbf{X}_{uv}, \mathbf{X}_{vv}$ are second partial derivatives.

Second derivatives

Compute:

$$\mathbf{X}_{uu} = \begin{pmatrix} -(R + r\cos v)\cos u \\ -(R + r\cos v)\sin u \\ 0 \end{pmatrix},$$

$$\mathbf{X}_{uv} = r\sin v \begin{pmatrix} \sin u \\ -\cos u \\ 0 \end{pmatrix},$$

$$\mathbf{X}_{vv} = -r \begin{pmatrix} \cos v\cos u \\ \cos v\sin u \\ \sin v \end{pmatrix}.$$

Dot products with the normal

Using $\mathbf{n} = (\cos u\cos v,\ \sin u\cos v,\ \sin v)^\top$ we obtain:

$$e = \mathbf{n}\cdot\mathbf{X}_{uu} = -(R + r\cos v)\cos v,$$

$$f = \mathbf{n}\cdot\mathbf{X}_{uv} = 0,$$

$$g = \mathbf{n}\cdot\mathbf{X}_{vv} = -r.$$

Matrix forms

Thus the second fundamental form reads

$$\mathrm{II} = \begin{pmatrix} e & f \\ f & g \end{pmatrix} = \begin{pmatrix} -(R + r\cos v)\cos v & 0 \\ 0 & -r \end{pmatrix}.$$

1.6 Principal curvatures, Gaussian and mean curvature

Since $F = f = 0$, the Weingarten (shape) operator is diagonal in the parameter basis and principal curvatures are simply

$$\kappa_1 = \frac{e}{E}, \qquad \kappa_2 = \frac{g}{G}.$$

Substituting $E = (R + r\cos v)^2$, $G = r^2$ and the second-form coefficients above yields

$$\boxed{\kappa_1 = -\frac{\cos v}{R + r\cos v}, \qquad \kappa_2 = -\frac{1}{r}}$$

(the sign depends on the chosen orientation of $\mathbf{n}$; magnitudes are often more relevant: $|\kappa_1| = \dfrac{|\cos v|}{R + r\cos v}$, $|\kappa_2| = \dfrac{1}{r}$).

Gaussian curvature K and mean curvature H are

$$K = \kappa_1\kappa_2 = \frac{\cos v}{r(R + r\cos v)},$$

$$H = \tfrac{1}{2}(\kappa_1 + \kappa_2) = -\frac{R + 2r\cos v}{2r(R + r\cos v)}.$$

Remark on orientation: If the opposite unit normal $-\mathbf{n}$ is chosen, all signs in the second fundamental form and principal curvatures change simultaneously.

1.7 Analytical integrals: surface area and volume

Using the surface element $dS = r(R + r\cos v)\,du\,dv$ we calculate the semi-torus area:

$$A_{\text{semi}} = \int_0^{2\pi}\int_0^{\pi} r(R + r\cos v)\,dv\,du = 2\pi r\int_0^{\pi}(R + r\cos v)\,dv$$
$$= 2\pi r\left(R\pi + r\int_0^{\pi}\cos v\,dv\right) = 2\pi^2 Rr,$$

because $\int_0^{\pi}\cos v\,dv = 0$.

For the volume: by Pappus' centroid theorem (or direct 3D integration), the full torus volume equals $2\pi^2 Rr^2$, so the semi-torus volume is

$$V_{\text{semi}} = \tfrac{1}{2}\cdot 2\pi^2 Rr^2 = \pi^2 Rr^2.$$

1.8 Theoretical error estimates for numerical integration

Let $f \in C^2(U)$ be an integrand on $U = [a,b]\times[c,d]$. We consider the composite midpoint rule on a grid with steps $h_u = (b-a)/N$, $h_v = (d-c)/M$.

1.8.1 1D midpoint recall

For a one-dimensional integral $I = \int_a^b f(x)\,dx$ the composite midpoint rule satisfies (there exists $\xi \in [a,b]$)

$$E_{\text{mid}}(f) = \int_a^b f(x)\,dx - h\sum_{i=0}^{N-1} f\Big(x_i + \tfrac{h}{2}\Big) = -\frac{(b-a)h^2}{24}f''(\xi),$$

hence the bound

$$|E_{\text{mid}}(f)| \le \frac{(b-a)h^2}{24}\,\|f''\|_{\infty,[a,b]}.$$

1.8.2 Extension to 2D (iterative approach)

Write the double integral as an iterated integral

$$I = \int_c^d \Big(\int_a^b f(u,v)\,du\Big)dv.$$

Applying the 1D midpoint rule with respect to u for each fixed v, and integrating the resulting 1D error bound over v, yields

$$\Big|\int_c^d E_u(v)\,dv\Big| \le \frac{(b-a)h_u^2}{24}\,(d-c)\,\|f_{uu}\|_{\infty,U}.$$

Similarly integrating first in v and then in u gives a contribution bounded by

$$\frac{(d-c)h_v^2}{24}\,(b-a)\,\|f_{vv}\|_{\infty,U}.$$

Summing these partial contributions we obtain the practical upper bound

$$\boxed{|E_{2D}| \le \frac{(b-a)(d-c)}{24}\Big(h_u^2\|f_{uu}\|_{\infty,U} + h_v^2\|f_{vv}\|_{\infty,U}\Big)}$$

which exhibits second-order convergence with respect to h_u and h_v.

1.8.3 Application to the surface-area integrand

For the area integrand $f(u, v) = r(R + r\cos v)$ we have $f_u = f_{uu} = 0$ and

$$f_v = -r^2 \sin v, \qquad f_{vv} = -r^2 \cos v.$$

Hence

$$\|f_{vv}\|_{\infty,U} = r^2, \qquad \|f_{uu}\|_{\infty,U} = 0.$$

Plugging into the 2D bound (with $a = 0, b = 2\pi, c = 0, d = \pi$) yields

$$|E_{2D}| \le \frac{(2\pi)\pi}{24}\, h_v^2\, r^2 = \frac{\pi^2 r^2}{12}\, h_v^2,$$

which shows that the error depends only on the v-discretization and scales quadratically with h_v.

1.9 Numerical algorithm (reproducible pseudocode)

```
Phi_midpoint(R, r, N, M):
    h_u = 2*pi / N
    h_v = pi / M
    sum = 0
    for i in 0..N-1:
        u = (i + 0.5) * h_u
        for j in 0..M-1:
            v = (j + 0.5) * h_v
            integrand = r * (R + r * cos(v))   # analytic integrand for dS
            sum += integrand
    A_num = sum * h_u * h_v
    # Error estimate using analytic bound on f_{vv}: max_f_vv = r^2
    E_est = (2*pi * pi)/24 * ( h_u^2 * 0 + h_v^2 * r^2 )
    return A_num, E_est
```

Practical grid choices. For engineering accuracy around 10^{-3}–10^{-4}, $N \in [40, 160]$, $M \approx N/2$ is typically sufficient. For 10^{-6} or better, use $N \ge 320$ or an adaptive quadrature in v.

1.10 Accuracy and numerical stability

Because the integrand is independent of u, refinement in v controls the accuracy. Increasing N beyond a moderate resolution will not reduce error substantially if M (or h_v) remains coarse. This explains observed empirical stabilization with respect to u-refinement.

1.11 Illustration of validation (what to include in the chapter)

Include:

1. A table with columns $N, M, A_{\text{num}}, A_{\text{err}}^{\text{abs}}, A_{\text{err}}^{\text{rel}}, V_{\text{num}}, \ldots$ for $N = 10, 20, 40, 80, 160, 320$ (with $M = N/2$).
2. A log–log plot of relative error vs. h_v (or vs. N) with an overlay line of slope -2 for comparison.
3. Short interpretation: verify the second-order behavior and discuss any deviations for small N.

1.12 Concluding remarks

This chapter provides a complete, rigorous derivation of the geometric quantities and a mathematically justified numerical protocol for verification. The provided error bounds make the numerical choices reproducible and explain the roles of the parameters R and r in accuracy.

Appendix A: Expanded algebraic computations and numerical script

Expanded algebra for cross product

Python numerical validation script

Below is a straightforward Python script (run with Python 3) that computes midpoint approximations of area and volume, prints a convergence table and writes CSV. Paste into a ‘.py‘ file and run.

```
# semi_torus_validation.py
import numpy as np
import pandas as pd

def numeric_area_volume(R, r, N, M):
    du = 2*np.pi / N
    dv = np.pi / M
    ui = (np.arange(N)+0.5)*du
    vj = (np.arange(M)+0.5)*dv
    U, V = np.meshgrid(ui, vj, indexing='ij')
    dS = r * (R + r * np.cos(V))
    A_num = np.sum(dS) * du * dv
    dV = (R + r * np.cos(V)) * r**2
    V_num = np.sum(dV) * du * dv
    return A_num, V_num

if __name__ == '__main__':
    R = 2.0
    r = 0.5
    A_analytic = 2 * np.pi**2 * R * r
    V_analytic = np.pi**2 * R * r**2

    Ns = [10, 20, 40, 80, 160, 320]
    rows = []
    for N in Ns:
        M = max(1, N//2)
        A_num, V_num = numeric_area_volume(R, r, N, M)
        rows.append({
            'N': N, 'M': M,
            'A_num': A_num, 'A_err_abs': abs(A_num - A_analytic),
            'A_err_rel': abs(A_num - A_analytic)/A_analytic,
            'V_num': V_num, 'V_err_abs': abs(V_num - V_analytic),
            'V_err_rel': abs(V_num - V_analytic)/V_analytic
```

```
    })
df = pd.DataFrame(rows)
print(df.to_string(index=False))
df.to_csv('semi_torus_convergence.csv', index=False)
```