Arquitetura do Simulador NPE-PSQ 2D

1. Visão Geral

O simulador NPE-PSQ 2D é um código de transporte magnetohidrodinâmico (MHD) que resolve as equações de evolução do plasma em geometria toroidal completa, usando coordenadas **radial (ρ)** e **poloidal (θ)**.

Evolução dos Modelos

Modelo	Dimensões	Variáveis	Complexidade	Fidelidade
0D	0 (global)	10 escalares	Baixa	Baixa
1D	1 (radial)	3 × 100 vetores	Média	Média
2D	2 (radial + poloidal)	3 × 100 × 64 matrizes	Alta	Alta
3D	3 (r + θ + ϕ)	3 × 100 × 64 × 32 tensores	Muito Alta	Muito Alta

O modelo 2D representa um **equilíbrio entre fidelidade física e custo computacional**, sendo adequado para:

- Estudos de instabilidades MHD (modos internos, ELMs)
- Análise de assimetrias poloidais
- Controle de posição vertical
- Simulação de disrupções

---

2. Sistema de Coordenadas

2.1 Coordenadas de Fluxo Magnético

Utilizamos o sistema de coordenadas **(ρ, θ, ϕ)** baseado em fluxo magnético:

ρ (rho): Coordenada radial normalizada `\rho = \sqrt{\frac{\Psi - \Psi_0}{\Psi_a - \Psi_0}}`

- ρ = 0: eixo magnético (centro)
- ρ = 1: borda do plasma (LCFS)

θ (theta): Ângulo poloidal

- $\theta \in [0, 2\pi]$
- $\theta = 0$: lado externo (low-field side)
- $\theta = \pi$: lado interno (high-field side)

ϕ (phi): Ângulo toroidal

- $\phi \in [0, 2\pi]$
- **Simetria toroidal:** $\partial / \partial \phi = 0$ (modelo 2D)

2.2 Geometria do Tokamak

Para uma seção poloidal com elongação (κ) e triangularidade (δ):

Raio maior: $R(\rho, \theta) = R_0 + \rho a \cos(\theta + \delta \sin\theta)$

Altura: $Z(\rho, \theta) = \rho a \kappa \sin\theta$

onde:

- $R_0 = 6.2$ m (raio maior)
- $a = 2.0$ m (raio menor)
- $\kappa = 1.7$ (elongação)
- $\delta = 0.33$ (triangularidade)

2.3 Métrica da Geometria

Jacobiano: $\mathcal{J} = \frac{\partial(R, Z)}{\partial(\rho, \theta)}$

Elemento de volume: $dV = 2\pi R \mathcal{J} \, d\rho \, d\theta$

---

3. Equações MHD 2D

3.1 Equações de Transporte

Temperatura Eletrônica: $\frac{3}{2} n_e \frac{\partial T_e}{\partial t} = -\nabla \cdot \mathbf{q}_e + Q_e$

Temperatura Iônica: $\frac{3}{2} n_i \frac{\partial T_i}{\partial t} = -\nabla \cdot \mathbf{q}_i + Q_i + Q_{ei}$

Densidade: $\frac{\partial n_e}{\partial t} = -\nabla \cdot \mathbf{\Gamma} + S_n$

onde:

- **q_e, q_i:** fluxos de calor

- **Γ:** fluxo de partículas
- **Q_e, Q_i:** termos fonte de aquecimento
- **Q_ei:** transferência colisional
- **S_n:** fonte de partículas

3.2 Fluxos de Transporte

Fluxo de calor: $\mathbf{q} = -n \chi_\perp \nabla_\perp T - n \chi_\parallel \nabla_\parallel T$

Fluxo de partículas: $\mathbf{\Gamma} = -D_\perp \nabla_\perp n - D_\parallel \nabla_\parallel n$

onde:

- **⊥:** perpendicular ao campo magnético
- **∥:** paralelo ao campo magnético
- **χ_⊥ << χ_∥** (anisotropia)
- **D_⊥ << D_∥**

3.3 Equação de Equilíbrio MHD

Equação de Grad-Shafranov: $\Delta^* \Psi = -\mu_0 R^2 \frac{dp}{d\Psi} - \frac{1}{2} \frac{dF^2}{d\Psi}$

onde: $\Delta^* = R \frac{\partial}{\partial R} \left(\frac{1}{R} \frac{\partial}{\partial R} \right) + \frac{\partial^2}{\partial Z^2}$

Esta equação determina o **fluxo poloidal Ψ(R,Z)** em equilíbrio.

3.4 Equação de Posição Vertical

Dinâmica vertical: $M \frac{d^2 Z}{dt^2} = F_z^{mag} + F_z^{control}$

onde:

- **F_z^mag:** força magnética (instabilidade vertical)
- **F_z^control:** força de controle (bobinas de feedback)

---

4. Discretização Espacial

4.1 Grade Computacional

Grade estruturada (ρ, θ):

- **N_ρ = 100:** pontos radiais

- **N_θ = 64:** pontos poloidais
- **Total: 6400 pontos**

Espaçamento:

- $\Delta\rho = 1/99 \approx 0.0101$
- $\Delta\theta = 2\pi/64 \approx 0.0982$ rad

4.2 Método de Diferenças Finitas

Derivadas radiais (2ª ordem): $\frac{\partial f}{\partial \rho}\bigg|_{i,j} \approx \frac{f_{i+1,j} - f_{i-1,j}}{2\Delta\rho}$

Derivadas poloidais (2ª ordem): $\frac{\partial f}{\partial \theta}\bigg|_{i,j} \approx \frac{f_{i,j+1} - f_{i,j-1}}{2\Delta\theta}$

Laplaciano: $\nabla^2 f \approx \frac{f_{i+1,j} + f_{i-1,j} + f_{i,j+1} + f_{i,j-1} - 4f_{i,j}}{\Delta\rho^2}$

4.3 Condições de Contorno

Centro (ρ = 0):

- Simetria: $\partial f/\partial \rho = 0$

Borda (ρ = 1):

- Dirichlet: f = f_edge

Poloidal (θ = 0, 2π):

- Periódica: f(ρ, 0) = f(ρ, 2π)

---

5. Integração Temporal

5.1 Método Implícito ADI

Usamos o método **Alternating Direction Implicit (ADI)** para resolver as EDPs 2D:

Passo 1 (direção ρ): $\frac{f^{n+1/2} - f^n}{\Delta t/2} = L_\rho(f^{n+1/2}) + L_\theta(f^n) + S^n$

Passo 2 (direção θ): $\frac{f^{n+1} - f^{n+1/2}}{\Delta t/2} = L_\rho(f^{n+1/2}) + L_\theta(f^{n+1}) + S^n$

onde:

- **L_ρ:** operador de difusão radial
- **L_θ:** operador de difusão poloidal
- **S:** termos fonte

Vantagens:

- Estável para grandes passos de tempo
- Resolve sistemas tridiagonais (eficiente)
- Precisão de 2ª ordem no tempo

5.2 Passo de Tempo Adaptativo

Critério CFL: $\Delta t \leq \min\left(\frac{\Delta\rho^2}{2\chi_\perp}, \frac{\Delta\theta^2}{2\chi_\parallel} \right)$

Passo típico: $\Delta t \approx 0.1\text{-}1.0$ ms

---

6. Termos Fonte 2D

6.1 Aquecimento por ECRH

Deposição localizada: $Q_{ECRH}(\rho, \theta) = \frac{P_{ECRH}}{V_{dep}} \exp\left(-\frac{(\rho - \rho_0)^2}{2\sigma_\rho^2} - \frac{(\theta - \theta_0)^2}{2\sigma_\theta^2} \right)$

Parâmetros:

- $\rho_0 = 0.4$ (posição radial)
- $\theta_0 = 0$ (lado externo)
- $\sigma_\rho = 0.15$, $\sigma_\theta = 0.3$

6.2 Aquecimento por NBI

Deposição assimétrica:

- Injeção tangencial: $\theta_{inj} \approx \pi/4$
- Penetração radial: decai com ρ

6.3 Aquecimento Ôhmico

Proporcional à densidade de corrente: $Q_{ohm}(\rho, \theta) = \eta_{||} j_\phi^2(\rho, \theta)$

---

7. Instabilidades MHD

7.1 Modos Internos (Sawteeth)

Critério de Kadomtsev:

- Ocorre quando $q(0) < 1$

- Reconexão magnética no raio q = 1
- Redistribuição rápida de T_e e n_e

Implementação:

- Detectar q(0) < 1
- Aplicar redistribuição instantânea

7.2 Modos de Borda (ELMs)

Critério de Peeling-Ballooning:

- Ocorre quando gradiente de pressão excede limite
- Ejeção de plasma da borda

Implementação:

- Monitorar ∇p na borda
- Aplicar perda de partículas/energia

7.3 Instabilidade Vertical

Crescimento exponencial: $\frac{d^2 Z}{dt^2} = \gamma_v^2 Z$

onde $\gamma_v \approx 100\text{-}1000\ s^{-1}$ (tempo de crescimento: 1-10 ms)

---

8. Diagnósticos 2D

8.1 Perfis Médios

Média poloidal: $\langle f \rangle_\theta(\rho) = \frac{1}{2\pi} \int_0^{2\pi} f(\rho, \theta) \, d\theta$

8.2 Assimetrias

Assimetria up-down: $A_{UD}(\rho) = \frac{f(\rho, \pi/2) - f(\rho, 3\pi/2)}{f(\rho, \pi/2) + f(\rho, 3\pi/2)}$

Assimetria in-out: $A_{IO}(\rho) = \frac{f(\rho, 0) - f(\rho, \pi)}{f(\rho, 0) + f(\rho, \pi)}$

8.3 Conteúdo de Energia

$$W = \frac{3}{2} \int_0^1 \int_0^{2\pi} n(\rho,\theta) T(\rho,\theta) \mathcal{J}(\rho,\theta) \, d\theta \, d\rho$$

---

9. Arquitetura de Software

9.1 Módulos Principais

Plain Text

```
simulator_2d/
├── src/
│   ├── geometry/
│   │   ├── tokamak_geometry_2d.py      # Geometria toroidal
│   │   ├── coordinate_mapping.py       # Mapeamento (R,Z) ↔ (ρ,θ)
│   │   └── jacobian.py                 # Cálculo do Jacobiano
│   ├── transport/
│   │   ├── transport_2d.py             # Coeficientes χ(ρ,θ)
│   │   ├── heating_sources_2d.py       # Termos fonte 2D
│   │   └── solver_adi.py               # Solver ADI
│   ├── mhd/
│   │   ├── equilibrium.py              # Equação de Grad-Shafranov
│   │   ├── instabilities.py            # Sawteeth, ELMs
│   │   └── vertical_control.py         # Controle vertical
│   ├── control/
│   │   └── nmpc_2d.py                  # NMPC adaptado para 2D
│   └── diagnostics/
│       └── diagnostics_2d.py           # Diagnósticos 2D
```

9.2 Classe PlasmaState2D

Python

```
class PlasmaState2D:
    def __init__(self, n_rho=100, n_theta=64):
        self.n_rho = n_rho
        self.n_theta = n_theta

        # Perfis 2D
        self.T_e = np.zeros((n_rho, n_theta))  # [keV]
        self.T_i = np.zeros((n_rho, n_theta))  # [keV]
        self.n_e = np.zeros((n_rho, n_theta))  # [10²⁰ m⁻³]

        # Quantidades globais
        self.Ip = 0.0      # [MA]
        self.Z_pos = 0.0   # [m]
        self.Z_vel = 0.0   # [m/s]
```

---

10. Validação

10.1 Comparação com Códigos de Referência

- **ASTRA:** código de transporte 1.5D
- **TRANSP:** código de análise experimental
- **JOREK:** código MHD não-linear 3D

10.2 Testes de Verificação

1. **Convergência de malha:** refinar N_ρ, N_θ
2. **Conservação de energia:** $\int dW/dt = P_{input} - P_{loss}$
3. **Simetria:** verificar assimetrias artificiais
4. **Equilíbrio MHD:** verificar força de Lorentz

---

11. Performance Computacional

11.1 Estimativas

Custo por passo de tempo:

- Solver ADI: $O(N_\rho \times N_\theta) \approx 6400$ operações
- Cálculo de fontes: $O(N_\rho \times N_\theta)$
- **Total: ~0.1-1 ms por passo**

Simulação de 10 segundos:

- 10000 passos ($\Delta t = 1$ ms)
- Tempo de parede: ~10-100 segundos
- **Speedup: 100-1000×**

11.2 Paralelização

Estratégias:

- OpenMP para loops em θ
- GPU para solver ADI (cuBLAS, cuSPARSE)
- MPI para múltiplas simulações (scan de parâmetros)

---

12. Roadmap de Desenvolvimento

Fase 1: Geometria e Grade (1-2 dias)

- Implementar geometria toroidal
- Criar grade computacional 2D
- Calcular Jacobiano e métricas

Fase 2: Solver ADI (2-3 dias)

- Implementar método ADI
- Testar com equação de difusão 2D
- Validar conservação de energia

Fase 3: Termos Fonte 2D (1 dia)

- Adaptar fontes de aquecimento para 2D
- Implementar assimetrias

Fase 4: MHD e Instabilidades (2-3 dias)

- Implementar equilíbrio de Grad-Shafranov
- Adicionar sawteeth e ELMs
- Controle vertical

Fase 5: Validação e Otimização (2-3 dias)

- Comparar com códigos de referência
- Otimizar performance
- Documentação

Tempo total estimado: 8-12 dias

Autor: Sistema NPE-PSQ

Data: Dezembro 2025

Versão: 1.0