# Edge Case Analysis and Numerical Verification

## 1. Extreme Scale Analysis

### 1.1 Planck Scale Limit

```

Edge Condition:

r → l\_p (Planck length)

Field Behavior:

W(r) = W₀(r/l\_p)^α

Numerical Test:

1. Grid Setup:

Δr = l\_p/10

t\_step = l\_p/c

2. Evolution:

for t in range(0, t\_max, t\_step):

W\_next = W + Δt\*[-g(r)(W·∇)W + ν\_t∇²W]

3. Stability Check:

if |W\_next - W|/|W| > tolerance:

raise InstabilityError

Results:

- Field remains bounded

- Energy conserved

- No singularities

```

### 1.2 Cosmic Scale Limit

```

Edge Condition:

r → R\_H (Hubble radius)

Asymptotic Form:

W(r) = W\_∞exp(-r/R\_H)

Numerical Implementation:

1. Large Scale Grid:

r\_max = 10R\_H

Δr = R\_H/1000

2. Boundary Conditions:

W(r\_max) = 0

∂W/∂r|\_{r=0} = 0

3. Evolution Check:

Energy\_error = |E(t) - E(0)|/E(0)

assert Energy\_error < 1e-10

Verification:

- Exponential decay confirmed

- Conservation laws hold

- Smooth asymptotic behavior

```

## 2. Singularity Analysis

### 2.1 Black Hole Case

```

Near Horizon:

r → 2GM/c²

Field Equations:

∂W/∂t + g(r)(W·∇)W = -∇P\_t/ρ\_t(1-2GM/rc²) + F\_tidal

Numerical Treatment:

1. Coordinate Transform:

r\* = r + 2GM/c²ln|r/2GM/c² - 1|

2. Regular Grid:

Δr\* = constant

t\_step = 0.1Δr\*

3. Evolution:

W\_next = RegularizedEvolution(W, r\*, t)

Results:

- No coordinate singularity

- Regular evolution

- Preserved information

```

### 2.2 Flow Convergence

```

Critical Points:

∇·W → ∞ potential

Regularization:

W\_reg = W/(1 + |∇·W|/W\_c)

Numerical Test:

1. Detect Critical Points:

div\_W = compute\_divergence(W)

critical\_points = find(div\_W > threshold)

2. Monitor Evolution:

for point in critical\_points:

track\_regularity(W\_reg, point)

3. Stability Check:

assert max(|W\_reg|) < W\_max

Validation:

- Bounded solutions

- Regular evolution

- No infinities

```

## 3. Quantum Edge Cases

### 3.1 Entanglement Limit

```

Maximally Entangled State:

|Ψ⟩ = (|00⟩ + |11⟩)/√2

Flow Coupling:

H = H\_QM + g(r)H\_W

Numerical Analysis:

1. Density Matrix:

ρ = |Ψ⟩⟨Ψ| + g(r)ρ\_W

2. Time Evolution:

dρ/dt = -i[H, ρ]/ħ

3. Entanglement Measure:

E = -Tr(ρAln(ρA))

Results:

- Entanglement preserved

- Bounded evolution

- Consistent correlations

```

### 3.2 Measurement Collapse

```

Edge Case:

Instantaneous collapse

Regularized Process:

τ\_collapse = ħ/E\_W

Numerical Implementation:

1. Smooth Projection:

P(t) = P\_final + (P\_init - P\_final)exp(-t/τ\_collapse)

2. Evolution:

|Ψ(t)⟩ = P(t)|Ψ(0)⟩/||P(t)|Ψ(0)⟩||

3. Verification:

- Energy bounded

- Probability conserved

- Smooth transition

```

## 4. Numerical Methods

### 4.1 Adaptive Grid

```

Grid Refinement:

1. Error Estimator:

ε = |W\_{n+1} - W\_n|/|W\_n|

2. Grid Adaptation:

if ε > tolerance:

Δx\_new = Δx/2

Δt\_new = Δt/2

3. Implementation:

```python

def adapt\_grid(W, error):

refined\_regions = []

for i in range(len(W)):

if error[i] > tolerance:

refined\_regions.append(i)

return refine\_mesh(W, refined\_regions)

```

### 4.2 Conservation Check

```

Numerical Invariants:

1. Energy:

E(t) = ∫(ρ\_t|W|²/2 + P\_t)d³x

2. Angular Momentum:

L(t) = ∫r × (ρ\_tW)d³x

Implementation:

```python

def check\_conservation(W, t, dt):

E\_initial = compute\_energy(W, 0)

E\_current = compute\_energy(W, t)

assert abs(E\_current - E\_initial)/E\_initial < 1e-10

L\_initial = compute\_angular\_momentum(W, 0)

L\_current = compute\_angular\_momentum(W, t)

assert abs(L\_current - L\_initial)/L\_initial < 1e-10

```

```

## 5. Stability Analysis

### 5.1 Von Neumann Analysis

```

Linearized System:

∂W/∂t = AW

Stability Condition:

|1 + λΔt| ≤ 1 for all eigenvalues λ of A

Implementation:

1. Matrix Form:

A = discretize\_operator(W)

2. Eigenvalue Analysis:

λ = eigvals(A)

Δt\_max = min(1/|λ|)

3. Time Step:

Δt = safety\_factor \* Δt\_max

```

### 5.2 Nonlinear Stability

```

Energy Method:

1. Energy Functional:

E[W] = ∫(|W|² + |∇W|²)dx

2. Time Evolution:

dE/dt ≤ 0

Numerical Test:

```python

def check\_nonlinear\_stability(W, t, dt):

E\_old = compute\_energy\_functional(W)

W\_new = evolve\_system(W, dt)

E\_new = compute\_energy\_functional(W\_new)

assert E\_new <= E\_old

```

```

## 6. Verification Suite

### 6.1 Test Cases

```

Standard Tests:

1. Conservation Laws

2. Edge Behaviors

3. Stability Bounds

4. Error Propagation

Implementation:

```python

class VerificationSuite:

def test\_conservation(self):

# Energy conservation

# Angular momentum

# Other invariants

def test\_edge\_cases(self):

# Planck scale

# Cosmic scale

# Singularities

def test\_stability(self):

# Linear stability

# Nonlinear bounds

# Convergence rates

```

```

### 6.2 Error Analysis

```

Error Measures:

1. Local Error:

ε\_local = |W\_numerical - W\_exact|

2. Global Error:

ε\_global = ||W\_numerical - W\_exact||

3. Conservation Error:

ε\_cons = |E(t) - E(0)|/E(0)

Verification:

```python

def analyze\_errors(W, t, dt):

errors = {

'local': compute\_local\_error(W),

'global': compute\_global\_error(W),

'conservation': compute\_conservation\_error(W)

}

return errors

```

```