

# s-ETS: Shock-Proof Rate Benchmark

## Executive Summary

**s-ETS (Shock Exponential Smoothing)** is an intelligent rate validation algorithm that separates normal market rates from disruption premiums (strikes, festivals, weather events). Unlike traditional moving averages that are slow to adapt, s-ETS provides instant, accurate rate benchmarking during market chaos.

---

## The Business Problem

### Standard AI Failure During Disruptions

Scenario	Standard ML Response	Business Impact
<b>Strike starts</b> → Rate jumps ₹40K → ₹80K	✗ Flags as anomaly, rejects invoice	Trucks don't come, factory stops
<b>Strike ends</b> → Rate drops to ₹40K	✗ Still approves ₹70K (lagging average)	Overpays for weeks, bleeds money

**Root Cause:** Traditional moving averages are **slow to adapt** - they smooth everything equally, including shocks.

---

## The s-ETS Solution

### Core Concept

Separates rate into **two components**:

$$\text{Total Rate} = \text{Base Level} + \text{Shock Premium}$$

Component	What It Represents	Update Speed
<b>Base Level</b>	Normal market rate (no chaos)	Slow ( $\alpha = 0.2$ )
<b>Shock Premium</b>	Disruption surcharge (strike/festival)	Fast ( $\delta = 0.8$ )

### The Key Insight

**During Strike:** "Base ₹40K + Shock ₹40K = ₹80K. APPROVED."

**Strike Ends:** Shock resets to 0 instantly. ₹80K now FLAGGED as 100% over.

---

## The Algorithm (Equation 8 from Research Paper)

### Mathematical Foundation

```

def validate_rate(current_rate, prev_level, prev_shock, is_disruption):
    # Step 1: Prediction (what we expect the price to be)
    predicted = prev_level + (prev_shock if is_disruption else 0)

    # Step 2: Error calculation
    error = current_rate - predicted

    # Step 3: Update Level (SLOW - ignores chaos)
    new_level = prev_level + (alpha * error) # alpha = 0.2

    # Step 4: Update Shock (FAST - only during disruption)
    new_shock = prev_shock + (delta * is_disruption * error) # delta =

    # Step 5: Calculate Benchmark
    benchmark = new_level + (new_shock if is_disruption else 0)

    # Step 6: Verdict
    variance_pct = (current_rate - benchmark) / benchmark * 100
    return "APPROVED" if abs(variance_pct) <= 5% else "FLAGGED"

```

## Parameters Explained

Parameter	Value	Role
$\alpha$ (alpha)	0.2	Level smoothing - learns base rate slowly, ignores noise
$\delta$ (delta)	0.8	Shock adaptation - absorbs disruption premium quickly
Tolerance	$\pm 5\%$	Variance threshold for approval

## Why Two Different Speeds?

- **Level ( $\alpha=0.2$ ):** Should NOT react to temporary spikes. Needs to track the true underlying market rate.
  - **Shock ( $\delta=0.8$ ):** MUST react quickly to capture the full disruption premium on Day 1.
- 

## Real-World Simulation

### Scenario: Maharashtra Transporter Strike (7 Days)

Day	Event	Submitted Rate	Base Level	Shock	Benchmark	Verdict
1	Normal	₹45,000	₹45,000	₹0	₹45,000	✓ APPROVED
2	Strike Starts	₹80,000	₹45,000	₹35,000	₹80,000	✓ APPROVED
3	Strike Active	₹82,000	₹45,400	₹36,600	₹82,000	✓ APPROVED
4	Strike Active	₹78,000	₹45,700	₹35,800	₹81,500	✓ APPROVED
5	Strike Ends	₹45,000	₹45,700	→ ₹0	₹45,700	✓ APPROVED

6	Post-Strike	₹80,000	₹45,700	₹0	₹45,700	<b>✗ FLAGGED (75% over)</b>
7	Normal	₹46,000	₹45,800	₹0	₹45,800	<b>✓ APPROVED</b>

### Critical Observation:

- Day 5: The moment "End Strike" is toggled, shock resets to ₹0
  - Day 6: Same ₹80,000 rate that was approved on Day 2 is now FLAGGED
- 

## Business Value Analysis

### Without s-ETS (Traditional ML)

Week	Scenario	Loss
Week 1 (Strike)	Rejected ₹80K invoices → Lost trucks	₹5L production loss
Week 2-4 (Post-Strike)	Approved ₹70K invoices (lagging benchmark)	₹30K overpay × 50 trucks = ₹15L
<b>Total</b>		<b>₹20L + loss per event</b>

### With s-ETS

Week	Scenario	Outcome
Week 1 (Strike)	Approved ₹80K with "Shock Premium" explanation	✓ No production loss
Week 2-4 (Post-Strike)	Instantly flagged ₹70K as 55% over benchmark	✓ No overpayment
<b>Total Loss</b>		<b>₹0</b>

### ROI Calculation

Annual strikes/disruptions: ~6 events  
 Loss per event (without s-ETS): ₹20L  
 Annual savings with s-ETS: ₹1.2 Cr+

---

## Indian Market Considerations

### Supported Disruption Types

Event Type	Typical Impact	Detection Method
Transporter Strike	1.8x - 2.2x	Manual toggle
Diwali/Holi Rush	1.3x - 1.5x	Calendar-based
Monsoon Disruption	1.2x - 1.4x	Weather API
Fuel Price Spike	1.1x - 1.3x	Fuel index

### Pre-configured Lanes

Lane	Normal Base Rate
Mumbai-Delhi	₹45,000
Chennai-Bangalore	₹28,000
Pune-Hyderabad	₹32,000
Ahmedabad-Jaipur	₹25,000
Kolkata-Lucknow	₹38,000
Mumbai-Pune	₹15,000
Delhi-Chandigarh	₹18,000

---

## API Reference

### Validate Rate

```
POST /api/shock/validate-rate
Content-Type: application/json
```

```
{
  "lane": "Mumbai-Delhi",
  "rate": 80000,
  "update_model": true
}
```

#### Response:

```
{
  "submitted_rate": 80000,
  "benchmark": 45000,
  "base_level": 45000,
  "shock_premium": 0,
  "variance_pct": 77.78,
  "verdict": "FLAGGED_HIGH",
  "explanation": "Rate is 77.8% ABOVE benchmark. Expected: ₹45,000, Got
}
```

### Toggle Disruption

```
PUT /api/shock/disruption
Content-Type: application/json
```

```
{
  "lane": "Mumbai-Delhi",
  "is_disruption": true,
  "event_type": "STRIKE"
}
```

### Get All Benchmarks

```
GET /api/shock/benchmarks
```

---

# Technical Architecture

## Files

File	Purpose
backend/ml/shock_ets.py	Core s-ETS algorithm implementation
backend/shock_routes.py	FastAPI REST endpoints
pages/ShockRateBenchmark.tsx	Frontend UI

## Database Tables

```
-- Current benchmark state per lane
CREATE TABLE rate_benchmarks (
    lane VARCHAR(100) PRIMARY KEY,
    current_level DECIMAL(12,2),
    current_shock DECIMAL(12,2),
    is_disruption BOOLEAN,
    updated_at TIMESTAMP
);

-- Disruption event log
CREATE TABLE disruption_events (
    id INT AUTO_INCREMENT PRIMARY KEY,
    event_type ENUM('STRIKE', 'FESTIVAL', 'WEATHER', 'OTHER'),
    region VARCHAR(100),
    start_date DATETIME,
    end_date DATETIME,
    is_active BOOLEAN
);

-- Validation audit trail
CREATE TABLE rate_validations (
    id INT AUTO_INCREMENT PRIMARY KEY,
    lane VARCHAR(100),
    submitted_rate DECIMAL(12,2),
    benchmark DECIMAL(12,2),
    verdict ENUM('APPROVED', 'FLAGGED_HIGH', 'FLAGGED_LOW'),
    validated_at TIMESTAMP
);
```

---

## Summary

**s-ETS is your insurance against AI failures during market chaos.**

Capability	Benefit
✓ Separates Base from Shock	Understands WHY rate is high
✓ Instant shock reset	No overpaying post-disruption
✓ Full audit trail	CEO-ready explanation

- ✓ One-click toggle Operations can activate instantly

**Bottom Line:** Replace "Rate Anomaly Detected" with "Approved: Base ₹40K + Strike Surcharge ₹40K"