VaR Backtesting and Risk Management

Week 7 — Financial Management: Volatility, Risk, and AI

Yi-Hao Lai

2026 Spring

Today's Roadmap

1

The Story: The Regulator Calls

The VolTech Challenge

Scenario: The Japan FSA wants a full backtesting report in two weeks.

They require: Kupiec test, Christoffersen test, and stress testing under extreme scenarios.

If we fail, the fund holds more capital — less money for investment returns.

The Core Question

Does our VaR model do what it claims? If we say 99% VaR is breached 1% of the time, can we prove it?

Today's Learning Objectives

By the end of this session, you will:

1. Define VaR backtesting and explain why regulators require it
2. Implement the Kupiec (POF) unconditional coverage test
3. Implement the Christoffersen conditional coverage test
4. Interpret the Basel traffic light system
5. Design and execute stress tests

2

The Logic of Backtesting

What Is VaR Backtesting?

Backtesting = comparing VaR forecasts against actual losses.

Define the **hit sequence**:

$$I_t = \begin{cases} 1 & \text{if } r_t < -\text{VaR}_t \quad \text{(breach)} \\ 0 & \text{otherwise} \end{cases}$$

A correct model's hit sequence must satisfy:

1. **Unconditional coverage**: proportion of breaches $= 1 - \alpha$
2. **Independence**: breaches should not cluster

Why Both Properties Matter

Model A: 3 breaches in 250 days, spread across Jan, Jun, Nov.

Breach rate $= 1.2\% \approx$ expected 1%.

$\rightarrow$ Passes unconditional coverage.

$\rightarrow$ Passes independence.

Model B: 3 breaches in 250 days, all in the same week during a crash.

Same breach rate $= 1.2\%$.

$\rightarrow$ Passes unconditional coverage.

$\rightarrow$ Fails independence (clustering!)

Key Insight

The Kupiec test catches wrong counts. The Christoffersen test catches clustering. You need **both**.

3

Kupiec Unconditional Coverage Test

Kupiec POF Test (1995)

Kupiec Likelihood Ratio

$$LR_{uc} = -2 \ln \frac{p^x (1-p)^{T-x}}{\hat{p}^x (1-\hat{p})^{T-x}} \sim \chi^2(1)$$

- $T =$ total out-of-sample days
- $x =$ number of breaches
- $p = 1 - \alpha =$ expected breach rate (e.g., 0.01)
- $\hat{p} = x/T =$ observed breach rate

Decision: Reject H_0 if $LR_{uc} > 3.841$ (5% level)

Kupiec Test: Practical Considerations

For $T = 250$ days and 99% VaR:

- Expected breaches: $250 \times 0.01 = 2.5$
- Approximate passing range: 0 to 6 breaches
- ≥ 7 breaches $\rightarrow$ likely rejection

Low Power Warning

With only 250 days, the Kupiec test cannot reliably distinguish between a 1% and a 2% breach rate. This is a fundamental limitation of small samples, not a bug in the test.

4

Christoffersen Conditional Coverage Test

Transition Matrix

Define transition probabilities: $\pi_{ij} = P(I_t = j \mid I_{t-1} = i)$

	$I_t = 0$	$I_t = 1$
$I_{t-1} = 0$	n_{00}	n_{01}
$I_{t-1} = 1$	n_{10}	n_{11}

Independence means: $\hat{\pi}_{01} = \hat{\pi}_{11}$

Clustering means: $\hat{\pi}_{11} > \hat{\pi}_{01}$

(Once a breach occurs, more are likely to follow.)

Christoffersen Test (1998)

Conditional Coverage Test

$$LR_{cc} = \underbrace{LR_{uc}}_{\text{coverage}} + \underbrace{LR_{ind}}_{\text{independence}} \sim \chi^2(2)$$

- LR_{uc}: Kupiec unconditional coverage (same as before)
- LR_{ind}: independence test using transition counts
- Combined: reject if $LR_{cc} > 5.991$

Interpretation

A model can **pass** Kupiec (right count) but **fail** Christoffersen (clustered

5

Basel Traffic Light System

Basel Traffic Light Framework

Zone	Breaches (250 d)	Interpretation	Multiplier
Green	0–4	Acceptable	$3.0\times$
Yellow	5–9	Questionable	3.4–$3.85\times$
Red	≥ 10	Rejected	$4.0\times$

⚠ Financial Impact

Red zone = capital multiplier increases from $3.0\times$ to $4.0\times$. For a fund with \$10B in risk exposure, this means **\$10B more capital** locked up — money that cannot be invested.

Yellow Zone Details

The yellow zone has graduated multipliers:

Breaches	Multiplier
5	$3.40\times$
6	$3.50\times$
7	$3.65\times$
8	$3.75\times$
9	$3.85\times$

Key insight: Banks are strongly motivated to stay in the green zone. VaR model quality has *direct financial consequences*.

6

Stress Testing

Stress Testing Framework

Backtesting = validates model under *normal* conditions
Stress testing = evaluates performance under *extreme* scenarios

Historical stress tests:

- 2008 Global Financial Crisis
- 2020 COVID-19 crash
- 2011 European debt crisis
- 1997 Asian financial crisis

Hypothetical stress tests:

- Sudden 5% market drop
- VIX spike from 15 to 80
- Simultaneous Asian market crash
- +300 bps interest rate shock

Interpreting Stress Test Results

For each scenario, compute:

$$\text{VaR Exceedance Ratio} = \frac{\text{Scenario Loss}}{\text{VaR}}$$

Scenario	Loss	VaR Ratio
GFC 2008 (worst day)	-9.47%	$3.3\times$
COVID-19 2020	-12.77%	$4.5\times$
Flash Crash 2010	-3.87%	$1.4\times$
Hypothetical -10%	-10.00%	$3.5\times$

VaR is not designed for crises. Stress tests quantify exposure *beyond* VaR.

7

Python Implementation

Step 1: VaR Backtest Setup

```python
import numpy as np, pandas as pd
import yfinance as yf
from arch import arch_model
from scipy import stats

sp500 = yf.download("^GSPC",
    start="2015-01-01", end="2024-12-31")
prices = sp500["Close"].squeeze()
returns = 100 * np.log(
    prices / prices.shift(1)).dropna()

window = 1000; alpha = 0.99
n_oos = len(returns) - window
print(f"Backtest: {n_oos} days")
print(f"Expected breaches: {n_oos*0.01:.1f}")
```

Step 2: Rolling GJR-GARCH VaR

```
var_bt = np.full(n_oos, np.nan)
for i in range(n_oos):
    train = returns.iloc[i:i + window]
    model = arch_model(train, vol="Garch",
        p=1, o=1, q=1, dist="t")
    res = model.fit(disp="off")
    fv = res.forecast(horizon=1)
    vol = np.sqrt(fv.variance.values[-1, 0])
    nu = res.params["nu"]
    q = stats.t.ppf(1 - alpha, df=nu)
    scale = np.sqrt((nu - 2) / nu)
    var_bt[i] = -(res.params.get("mu", 0)
                  + q * scale * vol)

breaches = returns.iloc[window:] < -var_bt
print(f"Breaches: {breaches.sum()}")
```

Step 3: Kupiec Test

```python
def kupiec_test(hits, alpha=0.99):
    T = len(hits); x = hits.sum()
    p = 1 - alpha; p_hat = x / T
    lr = -2 * (x * np.log(p / p_hat) +
        (T-x) * np.log((1-p) / (1-p_hat)))
    pval = 1 - stats.chi2.cdf(lr, df=1)
    return lr, pval

lr, pval = kupiec_test(breaches)
print(f"LR_uc = {lr:.3f}")
print(f"p-value = {pval:.4f}")
print("PASS" if pval > 0.05 else "FAIL")
```

Step 4: Christoffersen Test

```python
def christoffersen_test(hits, alpha=0.99):
    h = hits.astype(int).values
    n00=n01=n10=n11=0
    for t in range(1, len(h)):
        if h[t-1]==0 and h[t]==0: n00+=1
        elif h[t-1]==0 and h[t]==1: n01+=1
        elif h[t-1]==1 and h[t]==0: n10+=1
        else: n11+=1
    pi01 = n01/max(n00+n01,1)
    pi11 = n11/max(n10+n11,1)
    # ... compute LR_ind and LR_cc
    return lr_cc, pval

lr_cc, pval = christoffersen_test(breaches)
print(f"LR_cc = {lr_cc:.3f}, p = {pval:.4f}")
```

Step 5: Basel Traffic Light

```python
def basel_zone(n_breaches):
    if n_breaches <= 4:
        return "GREEN", 3.0
    elif n_breaches <= 9:
        mult = {5:3.4, 6:3.5, 7:3.65,
                8:3.75, 9:3.85}
        return "YELLOW", mult[n_breaches]
    else:
        return "RED", 4.0

last_250 = breaches.iloc[-250:]
zone, mult = basel_zone(last_250.sum())
print(f"Zone: {zone}, Mult: {mult}x")
```

Step 6: Stress Testing

```
scenarios = {
    "GFC 2008":      -0.0947,
    "COVID 2020":    -0.1277,
    "Flash Crash":   -0.0387,
    "Hypo -5%":      -0.05,
    "Hypo -10%":     -0.10,
}
current_var = var_bt[-1]
for name, loss in scenarios.items():
    ratio = abs(loss)*100 / current_var
    flag = "!!" if abs(loss)*100 > current_var \
        else ""
    print(f"{name:20s} {ratio:.2f}x {flag}")
```

8

Application: The FSA Report

FSA Backtesting Summary

Metric	Nikkei 225	S&P 500
Breach rate	1.4%	1.0%
Kupiec p-value	0.265	1.000
Christoffersen p	0.144	0.571
Basel zone	Green	Green

Conclusion: GJR-GARCH with t-distribution passes all regulatory requirements for both indices.

Key advantage: Model adapts daily — recovers from crisis shocks faster than Historical Simulation.

9

Key Takeaways

Six Things to Remember

1. **Backtesting validates VaR**: Compare predictions against actual losses using the hit sequence I_t
2. **Kupiec test**: Tests unconditional coverage — is the total breach rate correct? ($\chi^2(1)$)
3. **Christoffersen test**: Tests coverage *and* independence — are breaches clustered? ($\chi^2(2)$)
4. **Basel traffic light**: Green (0–4), Yellow (5–9), Red (≥ 10) with direct capital impact
5. **Stress testing complements backtesting**: Normal conditions vs. extreme scenarios — both required
6. **GJR-GARCH + t**: Consistently achieves green zone, outperforming simpler models

Mission 7: The FSA Backtesting Report

Deliverables

1. Rolling 99% VaR with GJR-GARCH(t) for Nikkei & S&P
2. Kupiec and Christoffersen tests for both indices
3. Basel traffic light classification (last 250 days)
4. Backtest visualization (returns + VaR + breaches)
5. Stress tests: 3 historical + 2 hypothetical scenarios
6. 300-word executive summary letter to the FSA

Bonus: Compare 3 VaR methods (HS, GARCH-normal, GJR-GARCH-t) and determine which performs best.

Next Week Preview

Week 8: AI in Finance

David: "QuantStar is pitching a pure ML risk system — random forests, LSTMs. They claim 50% better accuracy."

Dr. Lin: "Better accuracy by what measure? Against what benchmark? With what interpretability?"

Topics: Machine learning for risk, model interpretability, AI + GARCH hybrid approaches