Simulación de Monte Carlo — Movimiento Browniano Geométrico

¿Qué es una simulación de Monte Carlo?

Es una técnica que genera miles de escenarios posibles para el precio futuro de un activo, usando números aleatorios y un modelo matemático. A partir de esos escenarios se estiman probabilidades y valores esperados.

Modelo utilizado: Movimiento Browniano Geométrico (GBM)

El precio en cada paso de tiempo sigue la fórmula:

```
S(t) = S(0) × exp( (μ - ½σ²) × Δt + σ × √Δt × Z )
```

Donde:

- `S(0)` → Precio inicial del activo
- `μ` (mu) → Retorno esperado anual
- `σ` (sigma) → Volatilidad anual
- `Z` → Variable aleatoria normal estándar ~ N(0,1)
- `Δt` → Tamaño del paso de tiempo

Código Python

```python
import numpy as np
import matplotlib.pyplot as plt
import matplotlib.ticker as mticker

# ─────────────────────────────────────────────
# PARÁMETROS (modifícalos según tu activo)
# ─────────────────────────────────────────────
S0        = 100      # Precio inicial del activo
mu        = 0.10     # Retorno anual esperado (10%)
sigma     = 0.20     # Volatilidad anual (20%)
T         = 1        # Horizonte de tiempo en años
N         = 252      # Número de pasos (días de trading en 1 año)
M         = 1000     # Número de simulaciones
semilla   = 42       # Semilla para reproducibilidad

# ─────────────────────────────────────────────
# SIMULACIÓN
# ─────────────────────────────────────────────
np.random.seed(semilla)
```

```
dt = T / N
t  = np.linspace(0, T, N + 1)

Z             = np.random.standard_normal((N, M))
retornos      = (mu - 0.5 * sigma**2) * dt + sigma * np.sqrt(dt) * Z
incrementos   = np.exp(retornos)
trayectorias = np.vstack([np.ones(M) * S0, incrementos])
precios       = np.cumprod(trayectorias, axis=0)   # forma (N+1, M)

# ─────────────────────────────────────────────
# ESTADÍSTICAS FINALES
# ─────────────────────────────────────────────
precios_finales = precios[-1, :]

media    = np.mean(precios_finales)
mediana  = np.median(precios_finales)
std      = np.std(precios_finales)
p5       = np.percentile(precios_finales, 5)
p95      = np.percentile(precios_finales, 95)

# Probabilidad de pérdida
prob_perdida  = np.mean(precios_finales < S0) * 100
prob_ganancia = 100 - prob_perdida

print(f"Precio medio final      : ${media:.2f}")
print(f"Desviación estándar     : ${std:.2f}")
print(f"Percentil  5% (VaR 95%) : ${p5:.2f}")
print(f"Percentil 95%           : ${p95:.2f}")
print(f"P(pérdida)              : {prob_perdida:.1f}%")
print(f"P(ganancia)             : {prob_ganancia:.1f}%")

# ─────────────────────────────────────────────
# VISUALIZACIÓN
# ─────────────────────────────────────────────
fig, axes = plt.subplots(1, 2, figsize=(14, 5))
fig.suptitle("Simulación de Monte Carlo — GBM", fontsize=13, fontweight="bold")

# Trayectorias
ax1 = axes[0]
ax1.plot(t, precios[:, :200], alpha=0.15, linewidth=0.6, color="steelblue")
ax1.plot(t, np.mean(precios, axis=1),          color="red",    linewidth=2,
label="Media")
ax1.plot(t, np.percentile(precios, 5,  axis=1), color="orange", linewidth=1.5,
linestyle="--", label="P5%")
ax1.plot(t, np.percentile(precios, 95, axis=1), color="green",  linewidth=1.5,
linestyle="--", label="P95%")
ax1.axhline(S0, color="black", linewidth=1, linestyle=":")
ax1.set_xlabel("Tiempo (años)")
ax1.set_ylabel("Precio ($)")
ax1.legend()
ax1.grid(True, alpha=0.3)

# Distribución final
```

```
ax2 = axes[1]
ax2.hist(precios_finales, bins=60, color="steelblue", edgecolor="white",
alpha=0.8, density=True)
ax2.axvline(media, color="red",    linewidth=2,   label=f"Media: ${media:.2f}")
ax2.axvline(p5,    color="orange", linewidth=1.5, linestyle="--", label=f"P5:
${p5:.2f}")
ax2.axvline(p95,   color="green",  linewidth=1.5, linestyle="--", label=f"P95:
${p95:.2f}")
ax2.axvline(S0,    color="black",  linewidth=2,   label=f"S0 = ${S0}
({prob_perdida:.1f}% pérdida)")
ax2.axvspan(precios_finales.min(), S0, alpha=0.10, color="red")
ax2.set_xlabel("Precio final ($)")
ax2.set_ylabel("Densidad")
ax2.legend()
ax2.grid(True, alpha=0.3)

plt.tight_layout()
plt.savefig("montecarlo_resultado.png", dpi=150, bbox_inches="tight")
plt.show()
```

Interpretación de resultados

Métrica	Descripción
Precio medio final	Valor esperado del precio al final del horizonte
P5% (VaR 95%)	Peor precio en el 5% de los escenarios más adversos
P95%	Precio en el 5% de los escenarios más favorables
P(pérdida)	Probabilidad de que el precio termine por debajo del precio inicial
Zona roja	Región de pérdida en la distribución final

Notas importantes

> **Supuesto de normalidad:** Este modelo asume que los retornos siguen una distribución normal, lo cual es una simplificación. En la práctica, los retornos financieros suelen tener **colas más pesadas** (fat tails) y **asimetría**, por lo que distribuciones como la **t de Student**, **Pareto** o **Johnson SU** pueden ajustarse mejor a datos reales.