Stochastic Volatility Models Massively Parallel in R

Domokos Vermes

and

Min Zhao

WPI Financial Mathematics Laboratory

Black–Scholes–Merton vs. Reality

BSM Assumptions

- **Gaussian returns**
- **Constant volatility**

Market Reality

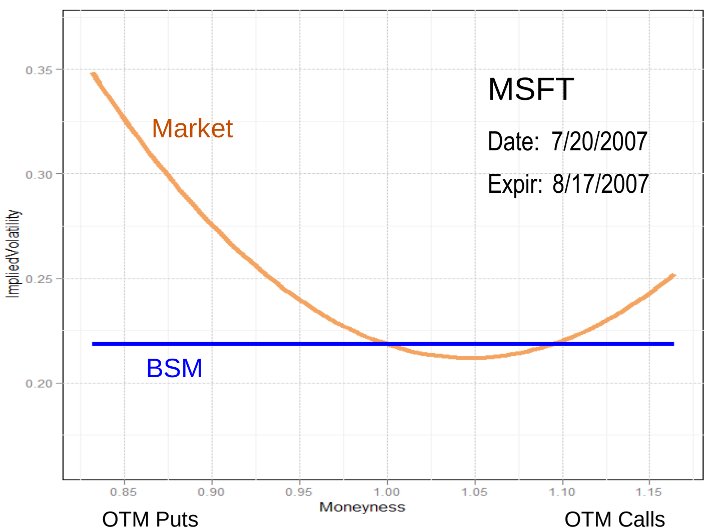
- **Non-zero skew**

  Positive and negative surprises not equally likely
- **Excess kurtosis**

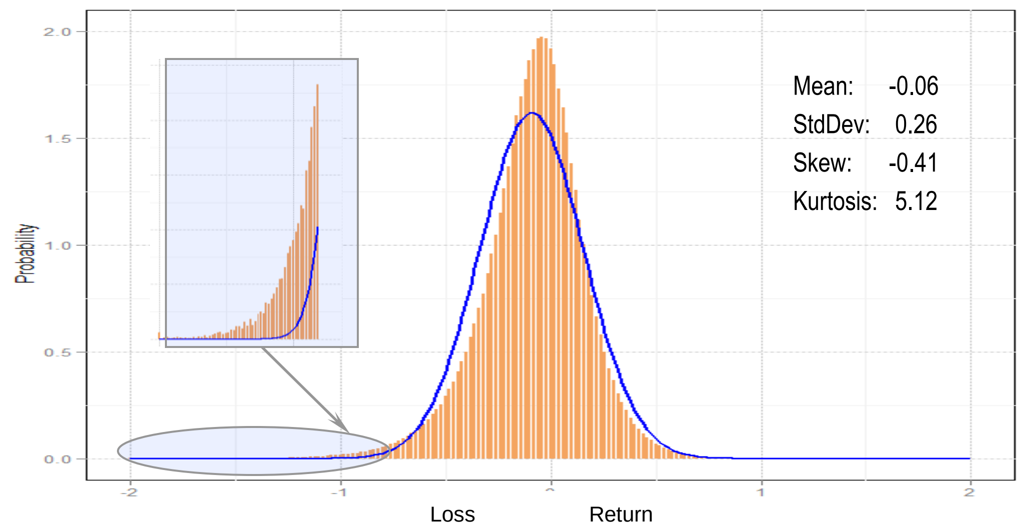
  Rare extreme events more frequent than in Gauss
- **Volatility smile**

  Out-of-the-money options are more expensive

Implied Volatility Smile



Heavy-tailed Return Distribution



Smile Consistent Market Models

Common cause of smile and non-normality

Non-constant (random) volatility (a.k.a. heteroscedasticity)

Volatility clustering No central limit theorem	⇒	More frequent extreme moves Heavy-tailed returns	⇒	Expensive insurance against extremes Implied volatility smile

Two-factor stochastic volatility (SV) model

Price $dS(t, \omega) = r \cdot S(t, \omega) \cdot dt + V(t,\omega)) \cdot S^{\beta}(t, \omega) \cdot dW(t, \omega)$

Volatility $dV(t, \omega) = \kappa \cdot (m-V(t, \omega))dt + \alpha(t, V(t, \omega)) \cdot dZ(t, \omega)$

Correlation $cor (W, Z) = \rho$

Given an SV model

volatility smile ⇔ return distribution

Proposed Approach

Market Volatility Smile ⇨ Implied Return Distribution

Approach

- Convert market option prices to implied volatilities
- Fit parameters of SV model to yield same smile as market
- Generate return distribution from fitted model

Advantages

- Based on market snapshot (no historical data needed)
- Options markets are forward looking
- Extracts info from options markets that are complementary to stock market

Challenges

- Inverse problem (non-trivial parameter sensitivities)
- Formula solutions exist only for 2-3 asymptotic models (Heston, SABR etc.)
- Monte Carlo deemed hopeless due to extreme compute intensity

Monte Carlo Model Fitting

Generate solution paths of 2-dim SDE

Paths must be sufficiently long (256~1024 steps) to induce volatility clustering

Simulate many paths and evaluate option prices

Very large number (> 1 million) of paths needed

Must include significant number of rare extreme events

Without enough extreme events smile doesn't bend

Calibrate model parameters to match market smile

Distance (objective) function non-convex, non-differentiable

Only function values (no derivatives) are available

- ›› Use robust Nelder - Mead optimizer (slow convergence)

Inverse problem (unpredictable parameter sensitivity)

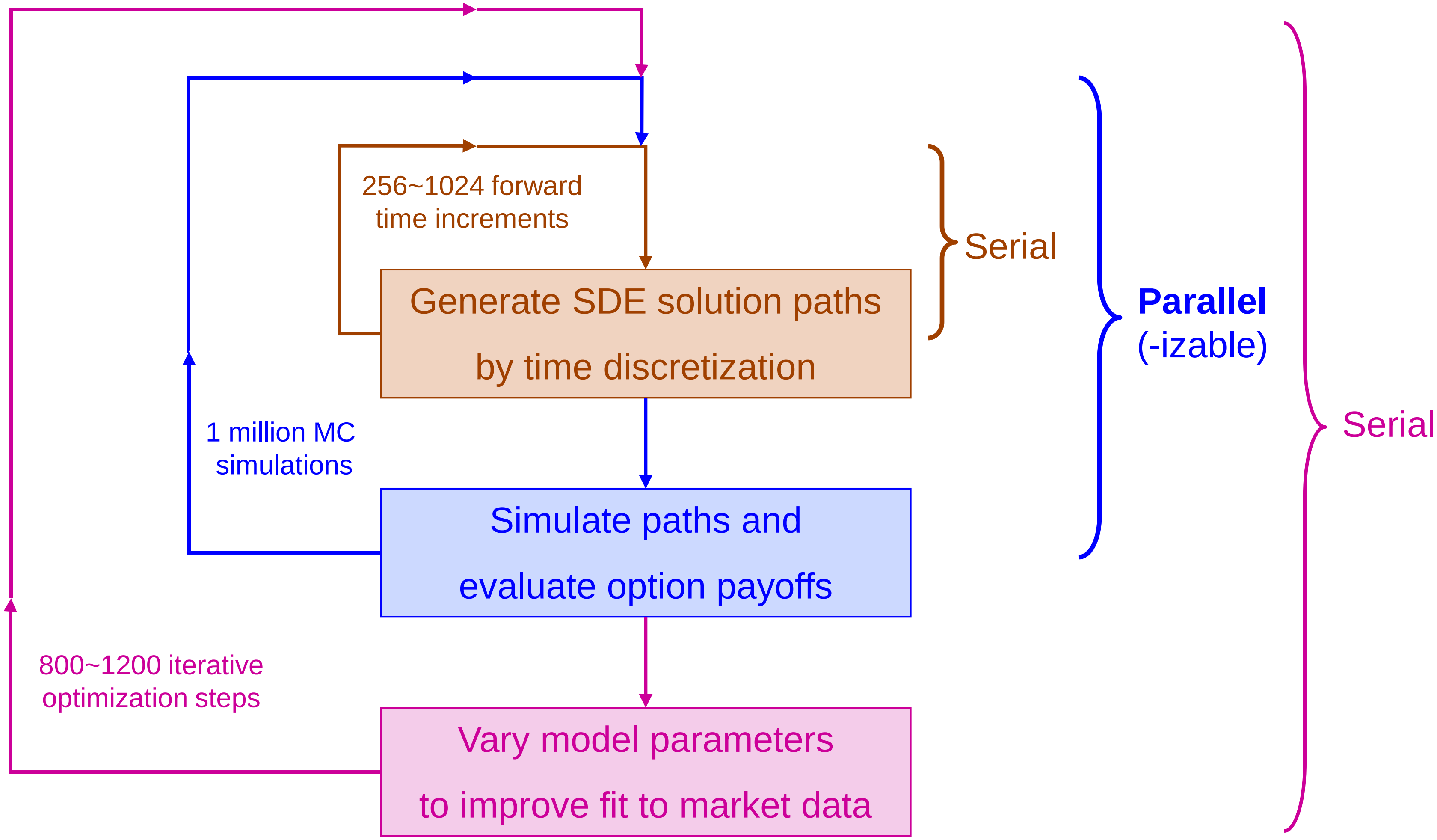
- ›› Provide “guidance” via penalty functions

Random increment requirement per model fit

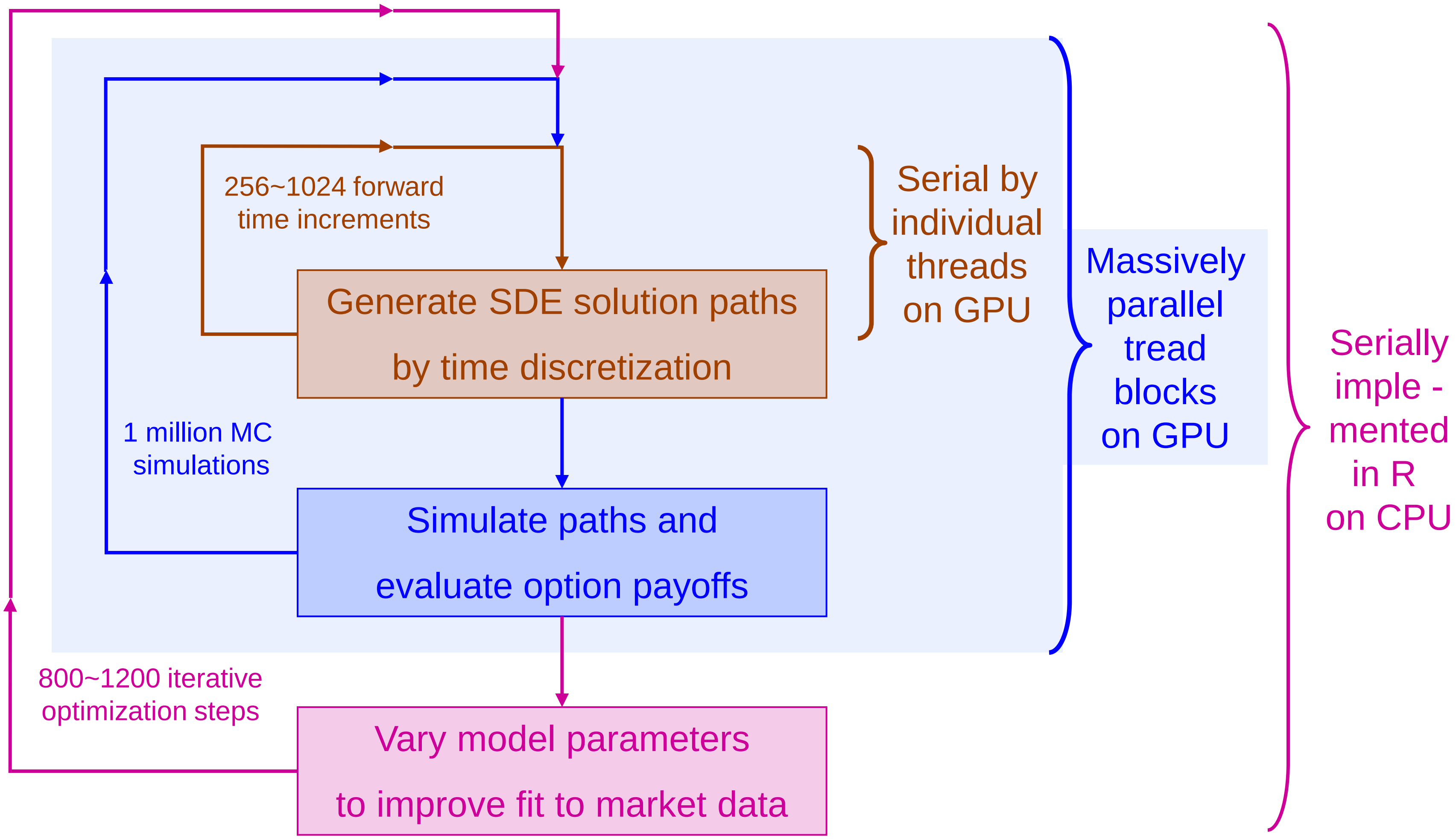
2 dims * 512 increments * 1M paths * 1000 optimization steps = **1 trillion**

(Literature is correct about extreme compute intensiveness of MC approach)

Paralellization of Model Fitting



Massively Parallel Implementation



CUDA Optimized Implementation

Execution organization

Single threads on individual cores

SDE solution trajectories, incl. necessary

Random number generation

Blocks of threads on multiprocessors

Parallel path generation and payoff evaluation

Blocks execute same operations data-parallel

Code optimization

- Saturate multiprocessors with waiting jobs
- Use ultra fast on-die shared memory efficiently
- Coalesce access to high-latency device memory
- Minimize transfers between device and host

Hardware-conscious code optimization is key to performance enhancement

Fermi 512-core GPU Architecture

Single Multiprocessor Organization

Hybrid Implementation in R

Executed on GPU in compiled CUDA-C

- Incremental SDE solution trajectory generation on individual GPU cores
- Simultaneous path simulation and option payoff evaluation by blocks cores

All parallelized CUDA functionality is wrapped in (C compiled) R functions

User benefits from parallel execution but doesn’t need to be aware of it

Executed on CPU in interactive R

- Data acquisition and organization
- Access and control functions of parallelized functionality
- Optimization steps of iterative model fitting
- Penalty function control of inverse problem
- Analysis and use of output from fitted model

Preserve and enjoy all interactive, graphical and statistical facilities of R

Benefits of Massively Parallel Approach

Acceleration by hybrid approach

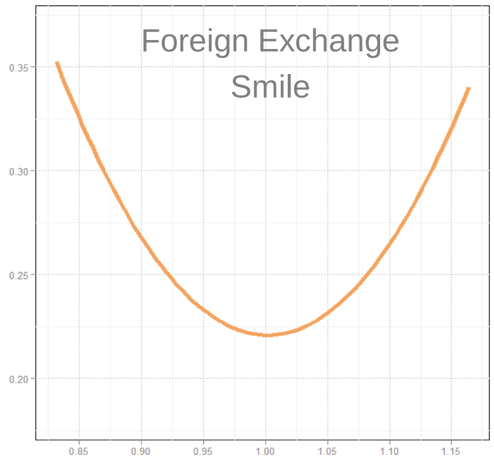
In R on CPU only: **75 hours** (> 3 days)
On CPU + GPU: **17 minutes**

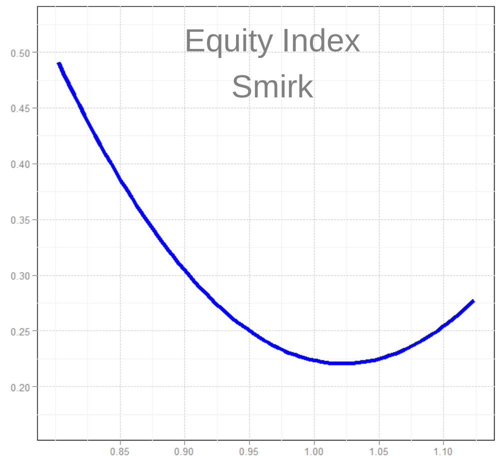
260x acceleration !

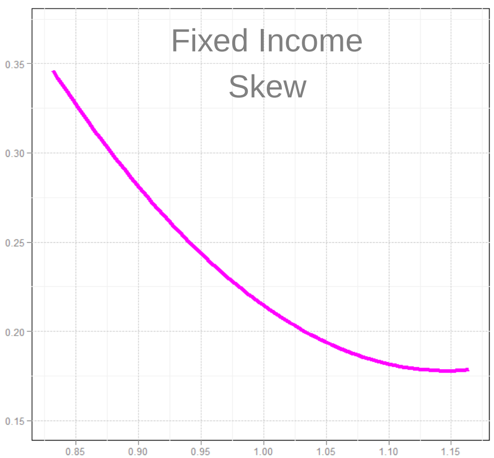
Makes possible
» Interactive analyses
» Trading desk use

Modeling flexibility

- Earlier only limited selection of SV models with formula solutions were feasible
  - » Heston, SABR, Fouque-Papanicoulau-Sircar
- Parallelized MC approach is feasible for arbitrary SDE based SV models
- Allows choice and comparison of models most suited to specific sub-areas







Financial Uses

Start

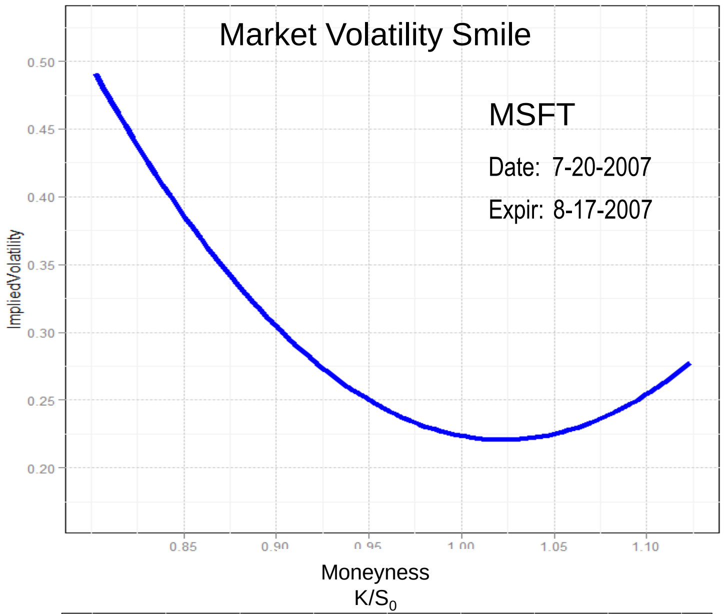
- Input implied volatility smile from option markets
  - » No historical data needed
  - » Forward looking views

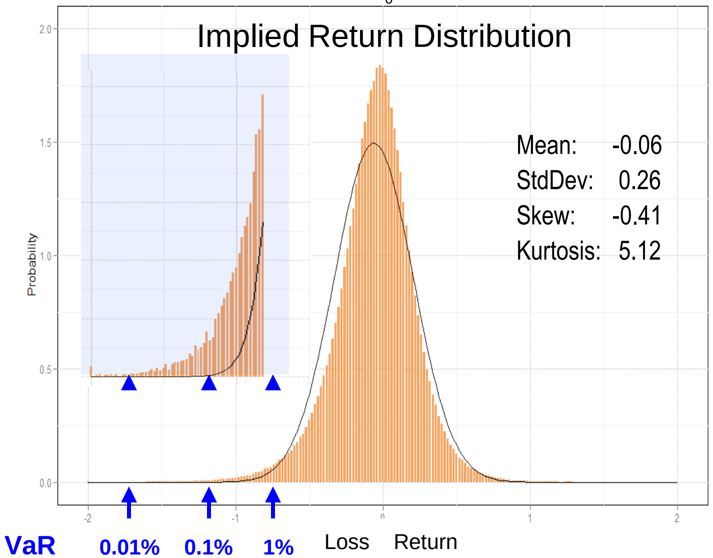
Fit

- Choose stochastic volatility model
- Calibrate parameters to market data
- Generate large sample from implied return distribution (IRD)

Use

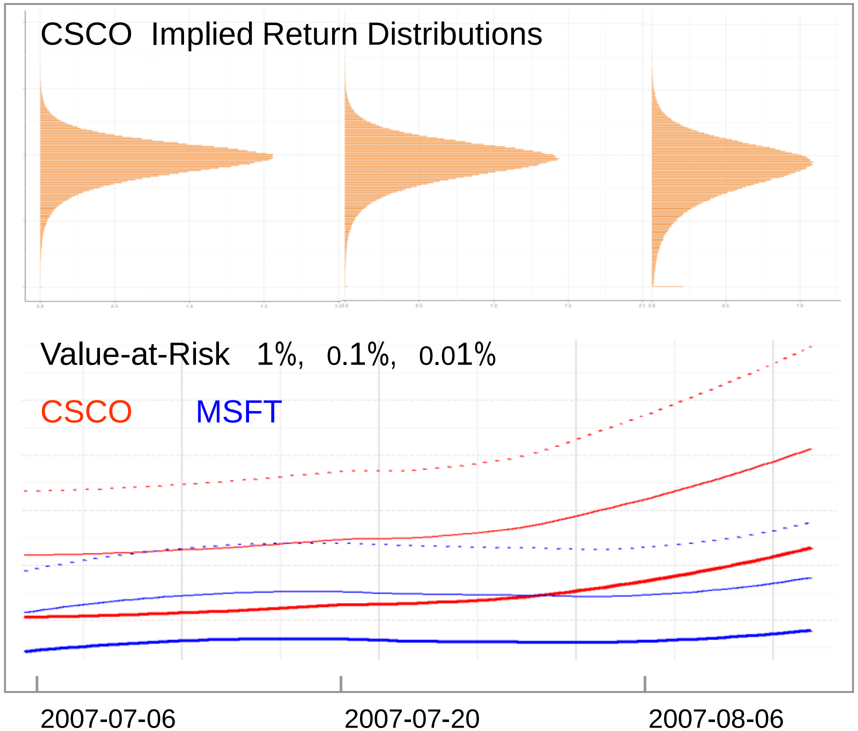
- Obtain risk measures VaR, ES from IRD
- Base smile-consistent pricing of other derivative securities on IRD
- Use insight offered by future returns as anticipated by options markets for
  - » Trading decisions
  - » Portfolio management





Sample Results

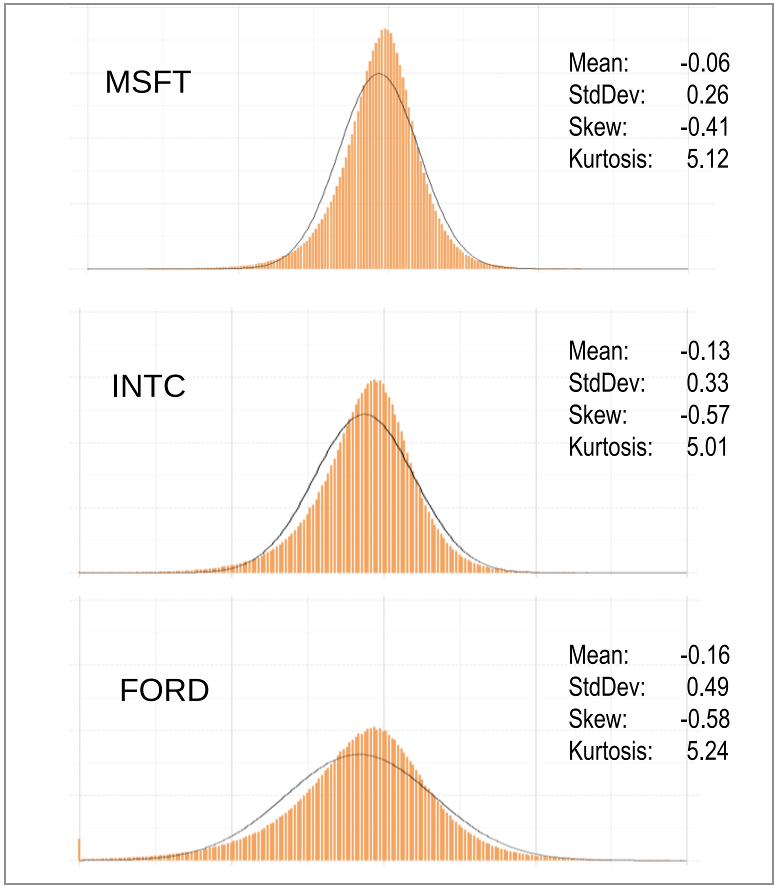
Evolution of risk perceptions



Stocks of both companies traded in a 3% range with no noticeable trend during the interval

Perception of increased risk for CSCO is complementary information that is not available from stock market observations

Relative performance anticipations



All relative performance data July 20, 2007

Summary

Integration of massively parallel simulations into R

Enables

- Implementation of models not previously feasible
- Use in real-time environments (trading desk, hedging quants)

Preserves

- Interactivity of R for exploratory analysis and experimentation
- Integration with graphical and statistical capabilities of R
- Traditional R programming paradigm
  - » No need to learn parallel programming

Remaining challenges

- Two-way CUDA-C ⇔ R interaction impractical
- Distribution on CRAN requires automatic compilation capability

Help or advice appreciated